



Defence Research and
Development Canada

Recherche et développement
pour la défense Canada



System Design - Commander HandhEld SyStem (CHESS)

M. Bélanger
N. Pageau
DRDC Valcartier

Defence Research and Development Canada – Valcartier

Technical Note
DRDC Valcartier TN 2013-303
May 2012

Canada

System Design - Commander HandhEld SyStem (CHESS)

M. Bélanger
N. Pageau
DRDC Valcartier

Defence Research and Development Canada – Valcartier

Technical Note

DRDC Valcartier TN 2013-303

May 2012

IMPORTANT INFORMATIVE STATEMENTS

- © Her Majesty the Queen in Right of Canada, as represented by the Minister of National Defence, 2012
- © Sa Majesté la Reine (en droit du Canada), telle que représentée par le ministre de la Défense nationale, 2012

Abstract

In a military context where commanders have to move regularly out of their offices, and other static or mobile Command and Control facilities, the need to remain connected is of critical importance. Therefore, mobile technologies are vital for allowing commanders to maintain access and interact with command support tools and processes necessary for them to exercise their authority under a wide range of operating conditions.

A prototype called CHESS – Commander HandhEld Support System, running on a wireless handheld device, was one of the important elements of a joint command decision support system for the 21st century (JCDS21). This prototype demonstrates how exchanges of information can occur between a wireless handheld device and Situational Awareness tools (COMMAND VIEW, Incident Management System), a planning system (Collaborative Operations Planning System) as well as decision support tools developed by JCDS21 TD. This report describes the main aspects of CHESS system design.

Résumé

Dans un contexte militaire, où les commandants sont sujets à œuvrer à l'extérieur de leurs bureaux ainsi que de leurs postes de commandement statiques ou mobiles, le besoin de rester connecté est d'une importance critique. En conséquence, les technologies mobiles sont d'une importance capitale afin de permettre aux commandants de maintenir un accès et une interaction avec les outils d'aide au commandement et aux processus nécessaires pour exercer leur autorité pour un large éventail de conditions opérationnelles.

Un prototype, appelé CHESS – Commander HandhEld Support System, fonctionnant sur un appareil de poche sans fil, était un des éléments important du système d'aide à la décision pour le vingt-et-unième siècle (JCDS21). Ce prototype démontre comment les échanges d'information peuvent être faits entre un appareil de poche sans fil et les outils d'éveil situationnel (COMMAND VIEW, Incident Management System), un système de planification (Collaborative Operations Planning System) ainsi que les outils d'aide à la décision développés dans le projet de démonstration technologique JCDS21. Ce rapport décrit les aspects principaux du design du système CHESS.

This page intentionally left blank.

Executive summary

System Design: Commander HandhEld SyStem (CHESS)

Normand Pageau; Micheline Bélanger; DRDC Valcartier TN 2013-303; Defence R&D Canada – Valcartier; May 2012.

Introduction or background: In a military context where commanders have to move regularly out of their office, and other static or mobile Command and Control facilities, the need to remain connected is of critical importance. Therefore, the requirement for mobile technologies aimed at allowing commanders to maintain access and interact with command support tools and processes necessary for them to exercise their authority under a wide range of operating conditions is essential.

Results: A prototype called CHESS – Commander HandhEld Support System, running on a wireless handheld device was one of the important elements of a joint command decision support system for the 21st century (JCDS21). This prototype demonstrates how exchanges of information can occur between a wireless handheld device and Situational Awareness tools (COMMAND VIEW, Incident Management System), a planning system (Collaborative Operations Planning System) as well as decision support tools developed by JCDS21 TD. This report describes the main aspects of CHESS system design

Future plans: CHESS is a first prototype aiming to present the information relevant to a commander on a wireless handheld device. Additional work will have to be invested in the identification of the most relevant information to be presented based on user's preference and role as well as the user's interface efficiency.

Sommaire

System Design: Commander HandhEld SyStem (CHESS)

Normand Pageau; Micheline Bélanger; DRDC Valcartier TN 2013-303; R & D pour la défense Canada – Valcartier; mai 2012.

Introduction ou contexte: Dans un contexte militaire, où les commandants sont sujets à œuvrer à l'extérieur de leurs bureaux ainsi que de leurs postes de commandement statiques ou mobiles, le besoin de rester connecté est d'une importance critique. En conséquence, les technologies mobiles sont d'une importance capitale afin de permettre aux commandants de maintenir un accès et une interaction avec les outils d'aide au commandement et aux processus nécessaires pour exercer leur autorité pour un large éventail de conditions opérationnelles.

Résultats: Un prototype, appelé CHESS – Commander HandhEld Support System, fonctionnant sur un appareil de poche sans fil, était un des éléments important du système d'aide à la décision pour le vingt-et-unième siècle (JCDS21). Ce prototype démontre comment les échanges d'information peuvent être faits entre un appareil de poche sans fil et les outils d'éveil situationnel (COMMAND VIEW, Incident Management System), un système de planification (Collaborative Operations Planning System) ainsi que les outils d'aide à la décision développés dans le projet de démonstration technologique JCDS21. Ce rapport décrit les aspects principaux du design du système CHESS.

Perspectives: CHESS est un prototype initial ayant pour but de présenter les informations d'intérêt au commandant sur un appareil de poche sans fil. Du travail additionnel sera nécessaire afin d'identifier l'information la plus pertinente à être présentée en fonction des préférences et des rôles des usagers ainsi que de l'efficacité de l'interface usager.

This page intentionally left blank.

Table of contents

Abstract	i
Résumé	i
Executive summary	iii
Sommaire	iv
Table of contents	vi
List of figures	viii
List of tables	xi
Acknowledgements	xii
1 Introduction.....	1
1.1 Context	1
1.2 Background.....	1
1.3 Scope	2
1.4 Document Overview.....	2
2 System Overview	3
2.1 Main System Requirements.....	3
Situation Awareness	3
Planning Process.....	3
Personal Information Management	4
Collaboration	4
2.2 Architecture Description	4
Main components interaction	5
Network	8
3 System Implementation	9
3.1 System Design Conditions	9
Technical requirements	9
JCDS 21 Project.....	9
SME PED Project.....	10
Project management considerations	10
3.2 User Interface	11
Overview	11
Specific techniques.....	15
3.3 Packages Structure and Breakdown	17
Tal5 Package.....	18
Login Package.....	22
Action Package	22
Bean Package	26

Utils Package.....	27
Exception Package	28
Listener Package	28
CV Package	29
ShowOpsInfosAction and ShowOpsInfoAction	30
ShowBriefingsAction	31
WelcomeAction	32
ShowBriefingAction	34
ShowBriefingResult	35
IMS Package.....	35
IMSAction 36	
COPlanS Package	39
COPlanS Bean Package	42
COPlanS Action Package.....	43
CoplansAction.....	45
InitiationAction	47
OrientationAction.....	49
CoaDevelopmentAction.....	51
WorkflowAction	53
Threats and ThreatAction.....	55
CCIRs and CCIRsAction	57
ShowDeliverableAction	59
EMPA Package.....	61
EmpaPlanExecutionListAction	62
4 Conclusion and Future work.....	65
References	66
Annex A ..Libraries list.....	67
List of symbols/abbreviations/acronyms/initialisms	71

List of figures

Figure 1: JCDS 21 Test Bed Architecture Overview	5
Figure 2: CHESS components interaction.....	6
Figure 3: CHESS network configuration.....	8
Figure 4: User interface layout	12
Figure 5: Application style configuration.....	13
Figure 6: Application labels edition	14
Figure 7: Browser specific tabular differences.....	15
Figure 8: Text summary interface	16
Figure 9: Contextual navigation	16
Figure 10: Incidents symbols.....	17
Figure 11 : Main application package breakdown.....	17
Figure 12 : The Model-View-Controller Architecture	19
Figure 13 : ta15 package breakdown.....	19
Figure 14 : TA15Action class.....	21
Figure 15 : login package break down.....	22
Figure 16 : action package breakdown.....	23
Figure 17 : TA15 login.jsp	23
Figure 18 : WelcomeAction dependencies	24
Figure 19 : TA15 welcome.jsp	24
Figure 20 : LogoutAction Dependencies.....	25
Figure 21 : ChangeCommandmentAction dependencies	26
Figure 22 : bean package breakdown	27
Figure 23 : utils package breakdown.....	27
Figure 24 : exception package breakdown	28
Figure 25 : listener package breakdown.....	28
Figure 26 : cv package breakdown.....	30
Figure 27 : ShowOpsInfosAction dependencies	31
Figure 28 : ShowOpsInfoAction dependencies	31
Figure 29 : ShowBriefingsAction dependencies	32
Figure 30 : cv briefings.jsp	32

Figure 31 : WelcomeAction dependencies	33
Figure 32 : cv welcome.jsp	33
Figure 33 : ShowBriefingAction dependencies	34
Figure 34 : cv briefing.jsp – display	34
Figure 35 : ims package breakdown	36
Figure 36 : IMSAction dependencies	37
Figure 37 : incident list.jsp	38
Figure 38 : incident detail.jsp	38
Figure 39 : incident list map.jsp	39
Figure 40 : incident map.jsp	39
Figure 41 : coplans package breakdown	42
Figure 42 : coplans bean package breakdown	43
Figure 43 : coplans action package breakdown	44
Figure 44 : coplans action dependencies	46
Figure 45 : coplans campaign list.jsp	46
Figure 46 : coplans campaign.jsp	47
Figure 47 : coplans plan.jsp	47
Figure 48 : InitiationAction dependencies	48
Figure 49 : coplans authorized movement.jsp	48
Figure 50 : coplans guidelines staff.jsp	49
Figure 51 : coplans initial reconnaissance.jsp	49
Figure 52 : OrientationAction dependencies	50
Figure 53 : coplans commander intent.jsp	50
Figure 54 : coplans commander coa guidance.jsp	51
Figure 55 : CoaDevelopmentAction dependencies	52
Figure 56 : coplans coa development.jsp (1)	52
Figure 57 : coplans coa.jsp	53
Figure 58 : WorkflowAction dependencies	54
Figure 59 : coplans workflow.jsp	54
Figure 60 : ThreatsAction and ThreatAction dependencies	55
Figure 61 : coplans threat list.jsp	56
Figure 62 : coplans threat.jsp	56
Figure 63 CCIRAction and CCIRAction dependencies	57

Figure 64 : coplans ccir list.jsp.....	58
Figure 65 : coplans ccir.jsp.....	58
Figure 66 : ShowDeliverableAction dependencies	59
Figure 67 : coplans initiation.jsp	60
Figure 68 : coplans orientation.jsp	60
Figure 69 : coplans coa development.jsp (2).....	61
Figure 70 : empa package breakdown	62
Figure 71 : EmplaPlanExecutionListAction dependencies	63
Figure 72 : empa empaplanexeclist.jsp.....	63
Figure 73 : empa empaplanexec.jsp	64

List of tables

Table 1: Description of main components.....	6
Table 2 : Main application packages description	18
Table 3 : ta15 packages description.....	20
Table 4 : TA15 jsp list.....	21
Table 5 : WelcomeAction user interaction.....	24
Table 6 : LogoutAction user interaction.....	25
Table 7 : ChangeCommandmentAction user interaction.....	25
Table 8 : cv jsp list.....	29
Table 9 : ShowBriefingsAction user interaction	31
Table 10 : WelcomeAction user interaction	33
Table 11 : IMSAction user interaction	34
Table 12 : ims jsp list.....	35
Table 13 : IMSAction user interaction	37
Table 14 : coplans jsp list	40
Table 15 : CoplansAction user interaction	45
Table 16 : InitiationAction user interaction.....	48
Table 17 : OrientationAction user interaction	50
Table 18 : CoaDevelopmentAction user interaction	51
Table 19 : WorkflowAction user interaction.....	53
Table 20 : ThreatsAction and ThreatAction user interaction	55
Table 21 : CCIRs and CCIRsAction user interaction.....	57
Table 22 : ShowDeliverableAction and ShowDeliverableResult user interaction.....	59
Table 23 : empa jsp list.....	61
Table 24 : EmpaPlanExecutionListAction user interaction.....	62
Table 25 : List of CHESS libraries.....	67

Acknowledgements

The authors want to thank JCDS21 scientific authority, the project director as well as project managers for their support during this project. We also want to thank Richard Moreau from Prolity Corporation as well as Stéphane Fortin from Fujitsu for their contributions to this R&D effort.

1 Introduction

1.1 Context

Commanders require the ability to exercise command regardless of where they are. This concept, also called “Command on the Move”, means that commanders are supported in achieving situational understanding, making decisions, disseminating directives, and following directives through execution in a designated HQ as well as on the road. The concept allows commanders to be virtually present at any required decision point. To support commanders on the move, enabling technologies will have to encompass communications capabilities, handheld mobile hardware handling of secure as well as unclassified information and mobile device applications providing access in real-time to current operational military C2 applications. DRDC Valcartier has developed an initial prototype called CHESS (Commander HandhEld Support System) to demonstrate how a wireless handheld device can be used to support a commander while away from a command post.[1]

The purpose of this document is to provide CHESS application system design information. The focus is on the key factors and technologies that influence its development and the overall decomposition of the software components. The level of detail is limited but ensures a basic level of understanding for a new participant to the project. The content of this report is mostly technical information aimed for a software development audience.

1.2 Background

DRDC Valcartier (DRDC-V) is responsible for managing a Technology Demonstration (TD) project: Joint Command Decision Support for the 21st Century (JCDS 21 TD), sponsored by Director Joint Force Capability (DJFC). The aim of JCDS 21 TD is to demonstrate a Joint Net-enabled, Collaborative Environment to achieve Decision Superiority. The main areas of expertise for this project are Command and Control (C2), human factors, human-computer interaction, information visualization, knowledge management, decision theories, artificial intelligence and experimentation. To achieve its objectives, the overall project is divided into five sub-projects:

- Front end analysis and experimentation;
- Organisational and individual factors;
- Situation awareness;
- Decision support;
- System integration and interoperability.

The decision support sub-project intends to investigate and demonstrate advanced decision-aid concepts to support the Joint Decision Making Process, considering that it is extremely difficult to obtain an omniscient understanding of all dimensions of the decision-making situation. The CHESS, which is part of this sub-project, aims to support commanders or senior staff to keep contact with their planning team, be aware of any new critical event and stay in touch with ongoing operations while away from their command post.

An initial study [4] has previously been completed to identify the information that would be most appropriate for a commander to exchange using a wireless device while away of its headquarters, considering the information available from the JCDS 21 systems. The second objective of that study was to investigate current wireless technologies, compare and select those that are the most suitable for the context of use. Following that study, a first concept of operation and a mock-up was produced to demonstrate and validate the approach. [2],[3]

1.3 Scope

This report presents CHESS as it was at the end of the JCDS21 TDP, in October 2008. It was written after one development cycle in a research and development (R&D) environment. This implied that a limited effort has been made on the design and a medium maturity reached in the related software. However, the resulting prototype uses recognized architecture and design methodology and is robust enough and well adapted to a Technology Demonstration Project (TDP).

For complementary information, it is suggested that the reader consults the preliminary version of the Software Requirement Specification (SRS) [2] and of the System Architecture and Requirements Allocation Description (SARAD) [3] documents and the JCDS High Level Functional Architecture [6].

1.4 Document Overview

The next section of this document describes an overview of the system by presenting some of the main requirements and a high level of the architecture. The third section is the core part of this report; it documents the development environment conditions, the user interface approach and all the detailed components of the architecture. The last section concludes this application design effort and proposes some improvement and directions for future work.

2 System Overview

2.1 Main System Requirements

The main requirement of the CHESS system is to allow commanders or senior staff to keep contact with their planning team, be aware of any new critical event and stay in touch with ongoing operations while away from their command post. The CHESS system does not intend to provide a complete virtual presence of the commander or to replace the current systems but to minimize disconnects and delays when he is on the move. For that purpose, the current application requirement focuses on four main aspects: situation awareness, planning process, personal information management and collaboration.

Situation Awareness

Situation awareness is currently enabled by existing systems. Accordingly, the CHESS system needs to accomplish two major functions, the first is to be able to retrieve situation awareness information from existing systems, namely Command View, IMS and EMPA. The second function is to be able to manage notification of important events. More specifically [2]:

- Access to Command View briefings and other important documents;
- Access to the incident list and details via IMS;
- Access to maps with geo-referenced operations or incidents;
- Ability to monitor plan execution and operations status via EMPA and other legacy C2 applications.
- Receive notifications related to the status of on-going planning activity;
- Receive notifications related to new incidents of interest;
- Receive notification on mission success criteria;
- Receive notification on decision points and associated triggers;
- Receive notification of availability of key planning products.

Planning Process

The planning process is the thinking process that leads to the decisions that will shape the plan. The commander plays a key role in that process by orienting its planning staff, giving directives and making decisions. In the context of this project, the focus is on the key commander tasks that could bring the planning process to halt when the commander is not present in the HQ. More specifically [2]:

- Monitor progress of planning cycles and associated key deliverables ;
- Access and control in planning battle rhythm;

- Participate in COA development as required or review the output of the COA Development stage of the OPP
- Access and/or add Commander Critical Information Requirements (CCIR);
- Consult the threats list;
- View and review the outputs of the planning process
- Contribute to the planning process by having the ability to:
- Complete missions analysis
- Craft a proposed mission statement
- Prepare and distribute Planning Guidance
- Draft a Concept of Operations and its associated Commander's Intent paragraph
- Review and approve key documents in particular orders of different formats

Personal Information Management

Personal information management is considered to be a core function of networked computer systems. Those functions should also be part of the mobile environment and the information should be synchronized with the desktop environment. This mainly includes [2]:

- Manage and synchronize contacts;
- Manage and synchronize the appointment schedule, calendar and meetings;
- Manage and synchronize tasks;
- Access past sent and received email messages.

Collaboration

Even if some collaborating requirements are already covered by commander participation in the planning process through the use of specialized functions, this section identifies more generic collaboration means that will allow the commander to keep in touch with his team.

- Send and receive email messages including document attachment;
- Text-based chat with other planning personnel, subordinate or superior commanders;
- Receive a full briefing presentation remotely;
- Simple audio call or teleconference with other personnel, subordinate or superior commanders.

2.2 Architecture Description

High level architecture of the CHESS application is quite dependent of the overall JCDS 21 Test Bed (TB). The main influence originates from the choice of a Service Oriented Architecture (SOA)

that relates on an Enterprise Service Bus (ESB). Conceptually, every core component and external additional system exposes its functionalities using a standard service definition and makes it available to other TB systems through the ESB. The ESB provides an abstraction layer on top of the SOA, acting as a message broker between applications. By giving an abstraction of the message format it may deal with services evolution by adapting message specific needs just by configuration changes instead of a recoding requirement. Figure 1 gives an overview of the TB architecture.

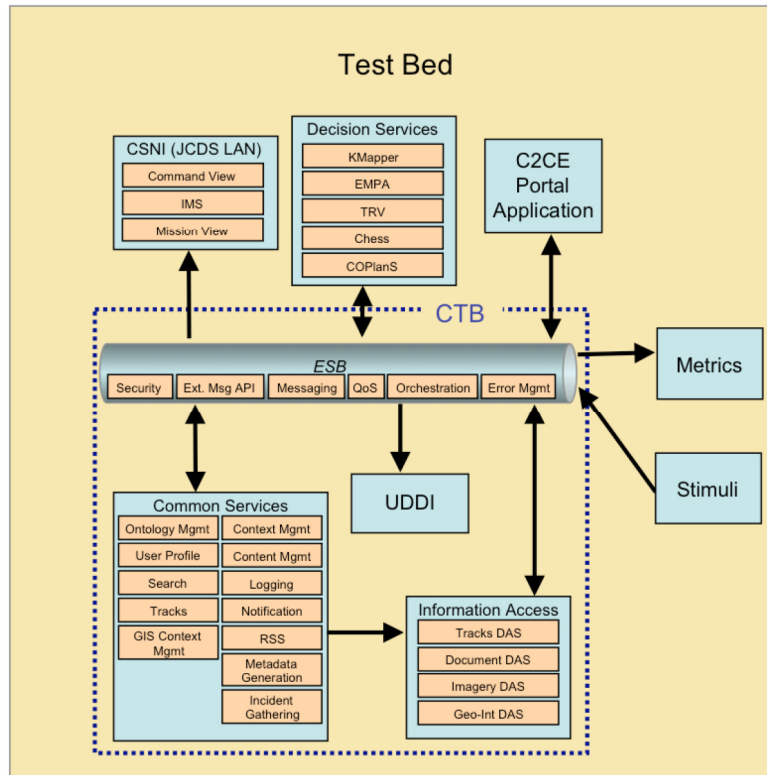


Figure 1: JCDS 21 Test Bed Architecture Overview

The combination of the ESB and the core services is designated as the Core Test Bed (CTB). In addition to the ESB, the CTB is mainly composed of two groups of services: Commons Services and Information Access Services. Moreover, a UDDI service is used by the ESB to allow discovery and integration of the services available on the system network. More details about those services and their implementation can be found in the CTB SAD [7].

The components external to the CTB are grouped in two main categories: those currently used on the operational CSNI baseline and those added by the JCDS 21 project as Decision Service. CHESS is part of that second category.

Main components interaction

The different tasks of the JCDS 21 project are identified with a TA number for management purposes. The identification of the development activity related to CHESS, represented as TA15,

has also been used for component identification. Without going through all the components of JCDS 21 test bed, this section presents those interacting with CHESS software as illustrated in Figure 2.

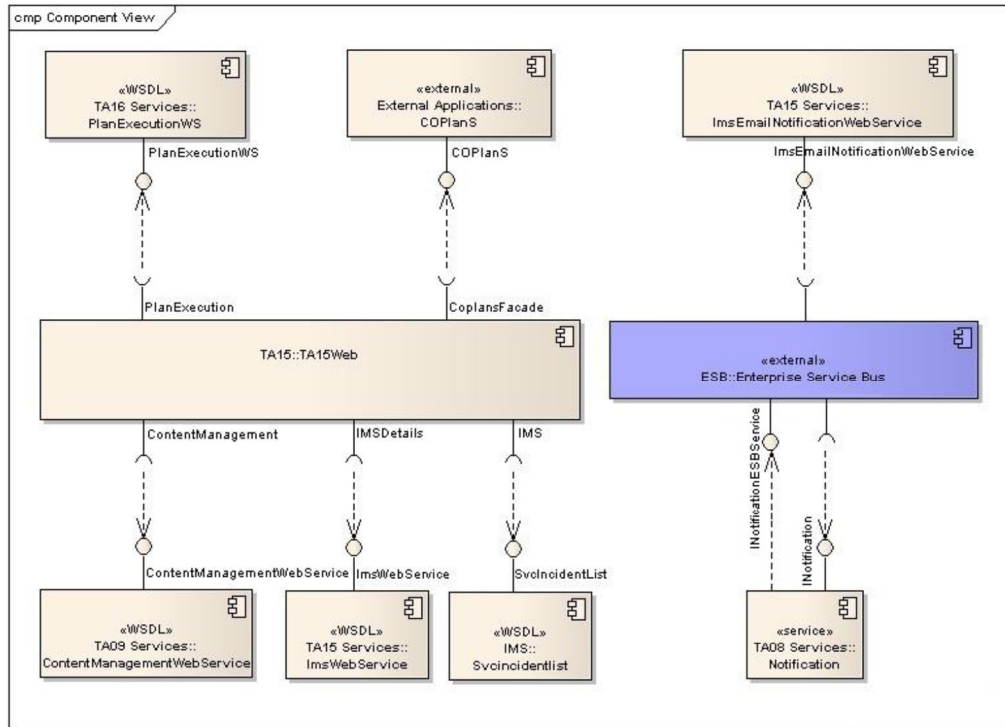



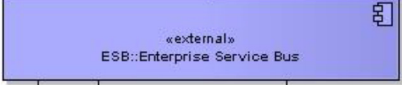


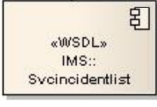



Figure 2: CHESS components interaction

The components on the diagram can be divided in two groups: those under CHESS development activity and those that are used by CHESS system. Table 1 gives a brief description of those components and their use in the context of CHESS.

Table 1: Description of main components

Component	Description
 <pre> «WSDL» TA16 Services:: PlanExecutionWS </pre>	This component is part of the EMPA project and provides service on plan execution information, mainly it status.
 <pre> «external» External Applications:: COPlanS </pre>	This component is part of COPlanS project. It provides the necessary libraries to access and manipulate COPlanS information.
 <pre> «WSDL» TA15 Services:: ImsEmailNotificationWebService </pre>	This component is part of CHESS project. It generates emails to notify the CHESS users when a new incident of interest occurs.

	<p>This component is provided by the open source software OpenESB. The ESB is configured using Business Process Execution Language (BPEL) to meet JCDS CTB requirements.</p>
	<p>This component is part of TA09 which aims to develop the some of the core components of the TB. The content management service allows the manipulation of a structured repository of documents. CHESSE essentially uses that service to retrieve important documents managed by Command View system (e.g. morning brief document).</p>
	<p>This component is part of the CHESSE project. A similar service was already provided by the IMS system but some key information for CHESSE was not available. This service has been temporarily developed as an extended IMS web service. It retrieves incident related information such as description and its location.</p>
	<p>This component is provided by the IMS system. It allows the retrieval of a filtered list of incidents managed by the IMS system.</p>
	<p>This component is part of TA08 development effort. It aims to create, generate and subscribe to different notifications. CHESSE subscribes to notifications of interest for the commander and generates emails with related information to be pushed to the handheld device.</p>

Network

The CHESS functionalities are highly dependent of the network availability. The schema illustrated in Figure 3 is an example of CHESS network configuration that has been built up for demonstration purposed. In that picture, JCDS 21 server contains the JCDS 21 TB that can be accessed through cellular or WI-FI communication. It is important to note that the communication security issues have not been addressed by this project.

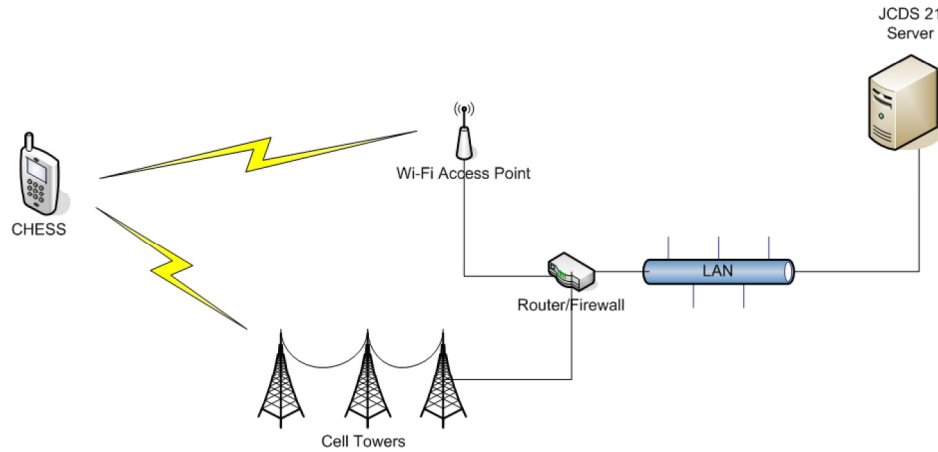


Figure 3: CHESS network configuration

3 System Implementation

3.1 System Design Conditions

This section puts more emphasis on the constraints and restraints that influence the design and the development phase of the CHESS application. Those conditions need to be described in order to better understand some of the decisions and some limits of the final product.

Technical requirements

First, as part of JCDS 21 project, the CHESS system has to be compliant with the architecture and the technology choices made for the overall JCDS 21 test bed. Secondly, the other major influence on the application design is its tight dependence on the Secure Mobile Environment Portable Electronic Device (SME PED) project. That project aimed to acquire the operational portable device where the CHESS application should be deployed. The following sub-sections describe some implications of those conditions.

JCDS 21 Project

JCDS 21 aims to build an integrated decision system of systems environment based on service oriented architecture (SOA). One key benefit of that type of architecture is the sharing and reusability of services in order to reduce the development effort and to facilitate the interoperability between systems. The two main requirements resulting from that architecture selection are:

1. Each generic need identified by the different systems must agree on the development and the use of a common service.
2. Each system component must be developed in a SOA approach in order to expose its functionalities to other systems to promote its interoperability.

At the user level, the JCDS 21 system intends to provide the user with integration transparency through a user interface homogeneity. To reach that goal, user interface guidelines [5] have been developed and imposed for every graphical component. Additionally, for web application development, it has been necessary to follow the standard from the “Common Look and Feel for the Internet 2.0” (CLF 2.0) [8] provided by the Government of Canada.

The environment configuration of the final demonstration of JCDS 21 is another element that influenced the application design. . The demonstration setup is a closed-loop environment that partially reproduces the operational systems network [9]. Since the email solution based on Apriva Sensa was not available on that setup, a Microsoft Exchange server has been configured and accessed through a WIFI connection. As the Defence Cryptographic Modernization Project could not provide the intended final hardware product that would run the CHESS application in time for the demonstration, it has been decided to use the device that was selected by the previous study [4]. Therefore, the testing of the application for the final technical adjustments were made on the HTC TyTN II.

From a technology point of view, some common development libraries were used to reduce the learning cost of the development team and ease the integration. Java was chosen as the primary language using MyEclipse as development platform.

Finally, main software systems like Oracle 10g and BEA Weblogic Server 10 were determined as core server components shared by the different applications of the JCDS 21 test bed to match as much as possible the software used in the operational environment.

SME PED Project

The Department of Defence has a capital project called the Defence Cryptographic Modernization Project (Defence CMP) which will ensure the availability of logistically supportable cryptographic devices, implementing robust cryptographic algorithms in a cost-effective manner throughout their life cycle. One of its sub-project was called the Secure Mobile Environment Personal Electronic Device (SME PED). It aimed to provide the first generation of secure Personal Digital Assistant (PDA) devices, which would provide truly mobile, secure access to the Departmental Wide Area Network (DWAN) and Consolidated Security Network Infrastructure (CSNI) networks using civilian cellular networks. The SME PED intended to interoperate with the Department's current inventory of Secure Telephones (using the Secure Communications Interoperability Protocol) and Network encryptors (using the High Assurance Internet Protocol Encryption (HAIPe) protocol).

One major requirement of the CHESS application is its ability to be used on the handheld device provided by the SME PED project. To comply with the specifications of that device the CHESS application has meet the following technical conditions:

- Must ensure that no software needs to be installed on the device;
- Must keep no information in memory;
- Must run on Windows CE 5.0;
- Must use QVGA display;
- Must be Web based and be executed within Internet Explorer 4.0;
- Must consider a low bandwidth environment.

Since it was not possible for CHESS development to get access to a simulation environment that exactly corresponds to the SME PED specification, an emulator of Windows Mobile 5 has been used.

Project management considerations

A number of conditions of the project management of JCDS 21 test bed development impact the CHESS application. While CHESS development was limited in time and budget, the time factor has been a critical issue. Because CHESS is mainly a data consumer application, it depends on data provider systems like Command View, IMS, EMPA, GIS service and Document Management service. Any delay in the development of those tools was directly influencing CHESS development timeline. Furthermore, the JCDS 21 test bed core services and its embedded sub-systems were developed in parallel, which impacted the development in two different ways. The first concerns

the fact that the development environment of the CHESS project was incomplete and needed to infer some components based on the architecture description. The second relates to the coordination effort required between the various sub-projects to develop a common understanding of similar concepts. By integrating data from independent systems that do not use a unique description, CHESS must deal with a single representation of these concepts. An example of this issue, the notion of “Operation” and “Plan” that has a different meaning in COPlanS, CV and EMPA systems. Following an understanding of the different definitions, fusion and translation techniques have been applied to the concepts to produce a coherent representation of the information.

Finally, the CHESS project itself had to deal with resource limitations since no expert on the development and graphic design for mobile device were available. Furthermore, the project development was based on only one main cycle since the end user community was available only at the final demonstration.

3.2 User Interface

This section gives a general overview of CHESS software graphical interfaces and looks at some techniques implemented to facilitate the user experience.

User interface is a critical aspect of software usability which is more significant with time sensitive systems where the user needs to operate the software in an environment under pressure. In systems designed for handheld devices, technical specifications of the device (such as available display size) greatly influence possible solutions. Those factors are part of the CHESS user interface design considerations that have led to the prioritization of two main objectives:

- Simple and efficient navigation by minimizing user interaction;
- Accurate selection of information to be displayed based on the user needs to minimize interface load.

For handheld systems, user interface experience is highly influenced by the graphical interface of the software as well as the hardware interaction. For example, it is highly recommended to have a touch screen interface and a full QWERTY keyboard to support user inputs. The CHESS user interface is the result of the application of the user interface guidelines that were developed for JCDS21 [5], as well as the structure of information that has been developed based on information requirements [1].

Overview

The CHESS user interface is designed for a handheld device with a basic web browser. It is composed of three main sections: the header, the navigation path and the interactive information display (see Figure 4). The screen aspect ratio is 4:3 with a portrait¹ QVGA resolution of 240 pixels wide and 320 pixels high. This aspect ratio is used in order to match as much as possible a normal reading page. The scrolling option is forced to the vertical scroll only to limit page panning and provide a more intuitive way for the user to read the information.

¹ QVGA “Portrait” alignment refers to a 240 × 320 display as opposed to “landscape” that refers to a 320 × 240 displays.

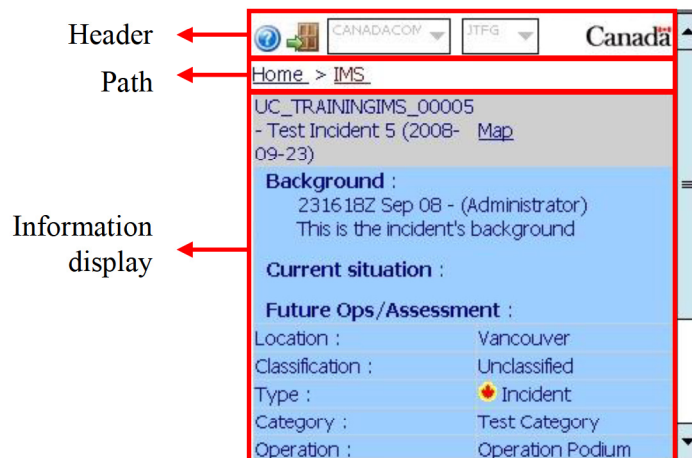


Figure 4: User interface layout

The “header” part of the screen is a static component of the interface and remains exactly the same for all the web pages of the application. It allows rapid access to the help feature of the application that gives contextual help on the current page, the logout feature that allows the user to exit the application at any time, the context configuration boxes that customize the information displayed based on the user profile and finally a logo image that can be used to set the identity of the application.

The “path” part of the screen gives the navigation path that has been accessed by the user to reach the current page. It can also be seen as an application contextual information bar. All the way points of the path are clickable links that can be used to quickly navigate back to one of these points.

The interactive “information display” part of the screen presents the user with the pertinent contextual information and provides the related functionalities of the application. In this section, the commander will access the information needed to support the decision making process and will have the possibility of injecting into the system some key directives, decisions, information or requests to his staff.

The look and feel of the graphical interfaces, as the fonts and colors, can be easily changed by editing the text style file “ta15-style.css” as shown in Figure 5.

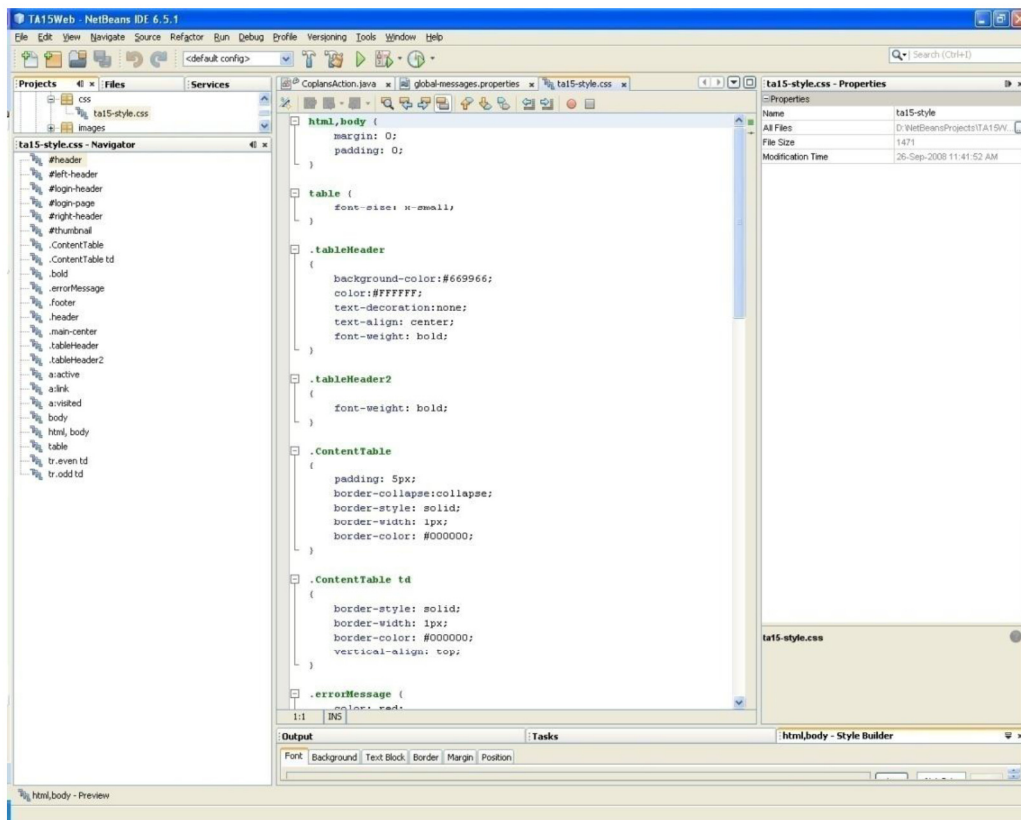


Figure 5: Application style configuration

Labels used in the application interface can be easily modified or even translated in a different language by editing the text file “global-messages.properties” as shown in Figure 6.

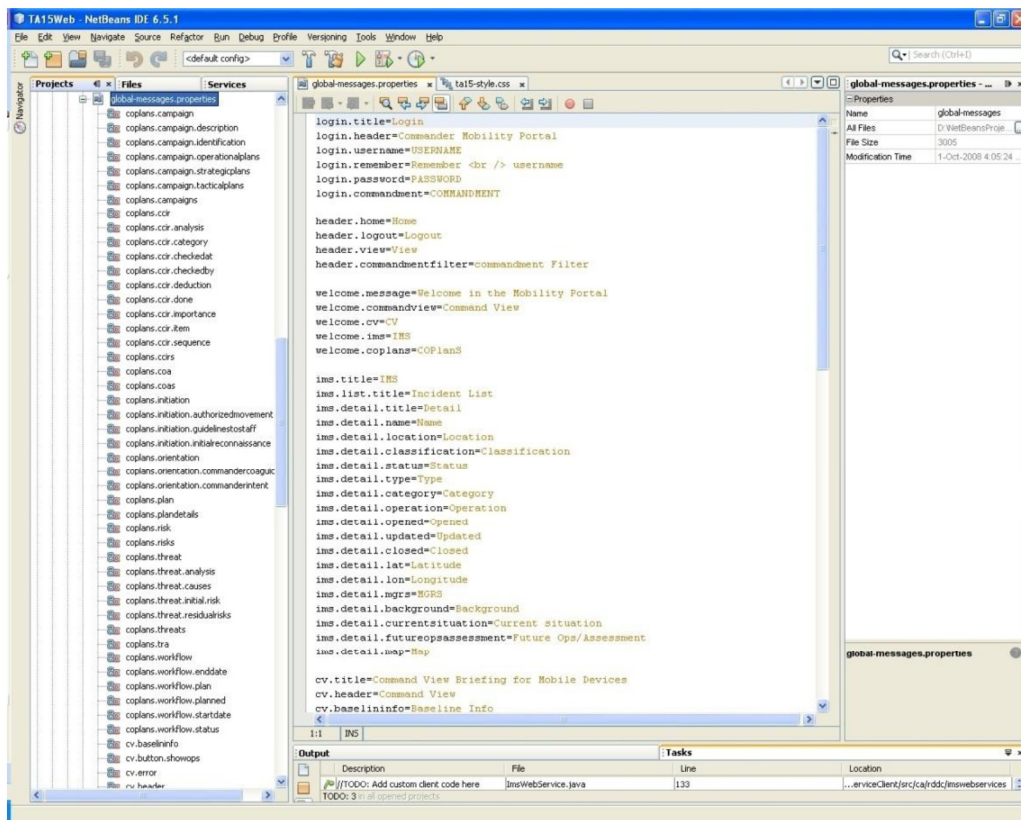


Figure 6: Application labels edition

As an emerging technology, browsers available on handheld devices don't have the same behavior when interpreting some basic HTML coding. An example is the tabular display that is intensively used by the CHESS software. As shown on Figure 7, the first image represents a browser's interpretation that lists the different column of the table while the second image displays the two columns next to each other (which represent the desired behavior).



Figure 7: Browser specific tabular differences

Specific techniques

Some more specific techniques have been used to reach the user interface objectives of the application. This section briefly discusses the most important ones.

One of the fundamental approaches of the interface design is to standardize the look and feel and operating mode of the application. This standardization allows the user to remain aware that many systems are involved in the application. In addition, it reduces the user's learning curve by reusing the same graphical components to execute the same type of activity.

In some cases, large text is associated with an information element and listing those elements with their associated text can take too much space and overload the display. Another technique is to limit the display to the list of elements and make them clickable to reach the detail about that element. However that solution may create a lot of undesired navigation if the element names are not clear enough. An alternative is to use a hybrid method that lists the elements with a small part of the associated text that gives a hint of the full content but does not overload the display, while keeping the element name clickable to get the full text content. An example of this technique is shown in Figure 8.





Figure 8: Text summary interface

In order to simplify the navigation between some elements and related key information from other systems or tools, the user interface provides contextual links between the related web pages in order to easily swap from one page to the other. The page illustrated in Figure 9 shows a “Plan Details” action available from an operation execution information page to quickly access the page that provided information on the plan itself.



Figure 9: Contextual navigation

In order to save some space, textual information can be replaced by small icons or symbol instead of enumerating a list of properties related to an element. However, that technique implies that the user is familiar with those icons and can easily infer the corresponding information.

A first example is  and  used in the header that allows to represent the “Help” and “Logout” action in very little space by using intuitive icons. This gain is really important considering that the header is present in all pages.

Another case is the use of acronyms combined with icons that are less intuitive but are known in the original system. One employment of this method is the “Threat and Risk Assessment” and “Commander Critical Information Requirements” that are long textual action labels that are replaced by ⚡ [TRA](#) and ⓘ [CCIRs](#).

Finally, a third case is the use of symbols as much as possible, as demonstrated in Figure 10. In this example, the maple leaf icon has a different meaning depending on the properties of the incident. Those properties are represented by different border and the filling colors.



Figure 10: Incidents symbols

3.3 Packages Structure and Breakdown

This section describes the package breakdown approach used for the development of the wireless handheld application. This development effort, as a subproject of JCDS Testbed system, is referred as “TA15” in the technical terminology. But, the whole project is under the “rdde” package illustrated in Figure 11 and also uses existing services in the JCDS Testbed environment.

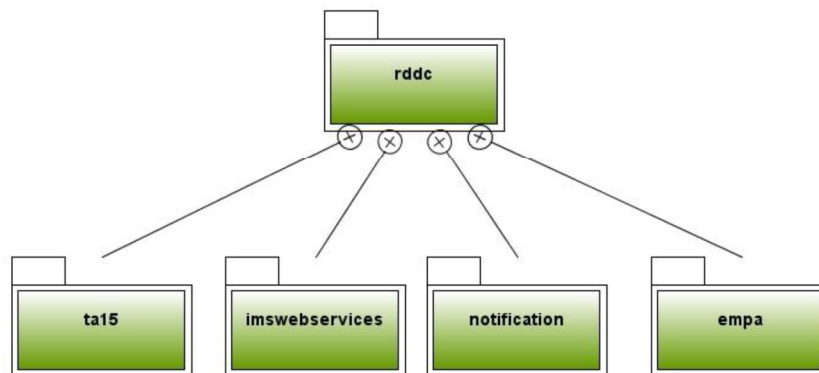


Figure 11 : Main application package breakdown

Table 2 : Main application packages description

Package Name	Description
ta15	This package contains all the development required by the wireless handheld application. The following sections will focus on its detail structure.
imswservices	This package contains the interfaces that allow the wireless application to get access to the incident management system (IMS) web service. This service is used to retrieve detailed information related to incidents.
notification	This package contains the necessary interface to handle notifications managed by the JCDS Testbed system and more particularly the event related to a plan status change.
empa	This package contains the interface to Execution Management and Plan Adaptation (EMPA) to get access to the list of plans that are monitored.

Ta15 Package

A common approach on the TA15 package and sub packages is to separate the model, the view and the controller as recommended by architectural design pattern MVC (http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/DEA2eTOC.html) and presented in Figure 12. The application model is composed of the bean package that contains the business objects and the client package contains the business logic. The action package is in charge of the application behavior by implementing the controller role. The handheld web application uses Java Server Pages (JSP) to render the view part of the design pattern.

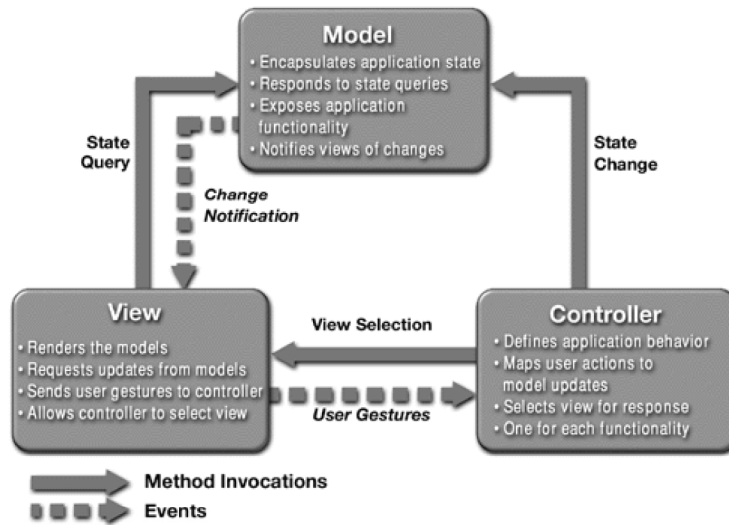


Figure 12 : The Model-View-Controller Architecture

The main package of the wireless handheld application is “ta15” that could be divided in two main categories: the package generic to the application and the packages related to the tools that the application supports (Figure 13).

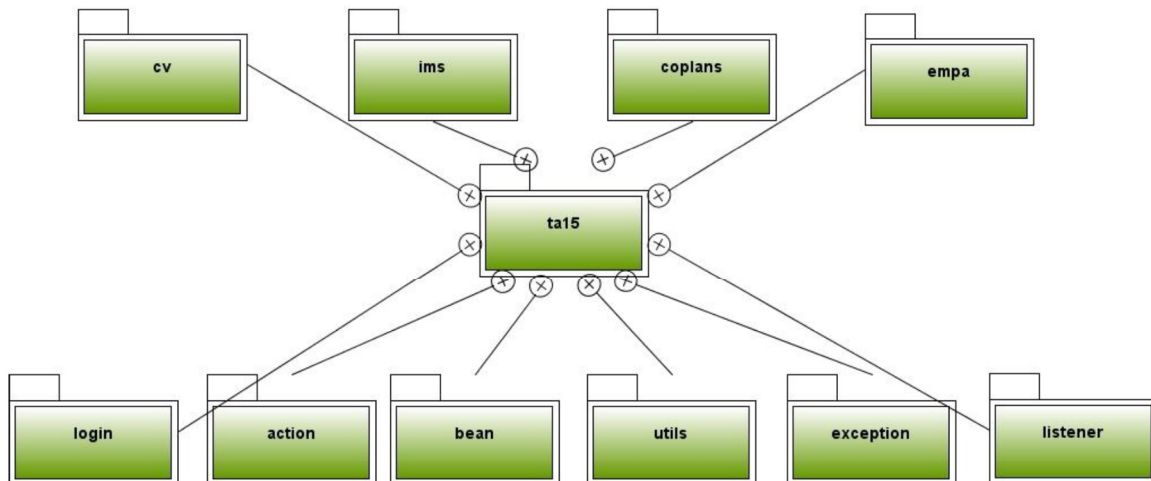


Figure 13 : ta15 package breakdown

Table 3 : ta15 packages description

Package Name	Description
cv	This package contains Command View (CV) system functions and related information managed by the application. The main capability is to access briefings and other important documents to the commander.
ims	This package contains Incident Management System (IMS) functions and related information managed by the application. The main capabilities are to retrieve and display incidents from IMS and maps from the geographic information system (GIS).
coplans	This package contains Collaborative Operation PLANning System (COPlanS) functions and related information managed by the application. The main capabilities are to give access to the commander to the operation planning process related information and to input directives and guidelines.
empa	This package contains Execution Management and Plan Adaptation (EMPA) functions and related information managed by the application. The main capability is to retrieve and to display plans monitored by the EMPA tool.
login	This package contains the functions needed to manage and proceed to the user authentication in the application. It currently uses the COPlanS server authentication mechanism.
action	This package contains the functions generic to the application and provides the common interface for implementing all the actions available through the application. It provides basic functionalities, mainly handling the HTTP requests and responses and contains the necessary methods to centralize the coordination and the information sharing between actions from different system packages (Figure 14).
bean	This package contains the generic data maintained by the application, mainly the application user information.
utils	This package contains the generic utility functions for the application.
exception	This package contains the functions needed for error management in the application.
listener	This package contains the functions that maintain users' sessions on the server.

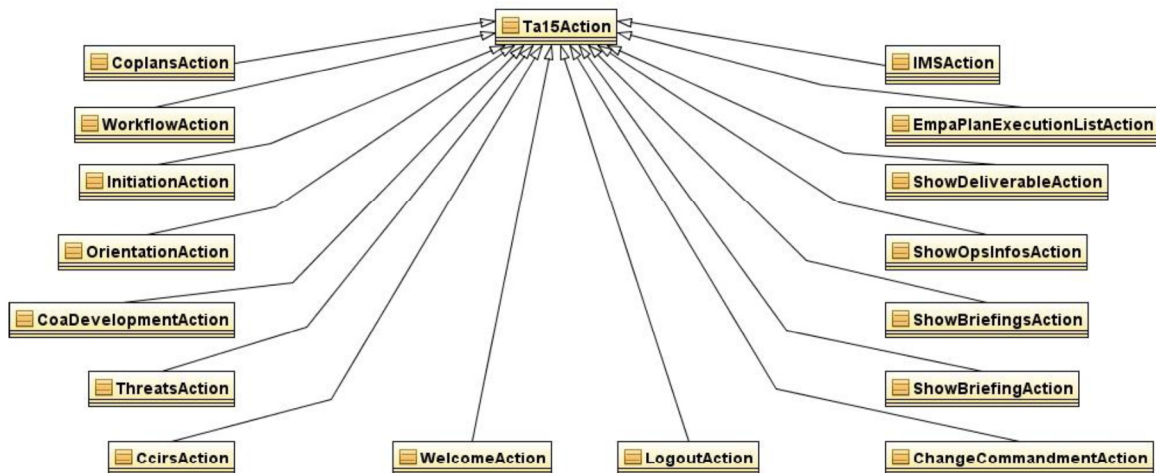



Figure 14 : TA15Action class

The global Java Server Pages for the application are listed and described in Table 4. The following sub-sections present action classes implemented with their dependencies on the business objects and shows an example of the view presentation.

Table 4 : TA15.jsp list

Web Application Structure	JSP	Description
	header.jsp	Display the header of the application with the global actions: Help, Logout and Change of commandment.
	login.jsp	Display login page of the application. It allows user to enter its username and password.
	logout.jsp	Display the temporary logout page while redirected to the login page.
	redirect.jsp	Display the temporary page while the application needs to be redirected to the starting page.
	welcome.jsp	Display the starting page of the application.

Login Package

This package contains the functions needed to manage and proceed to the user authentication for the application (Figure 15). The login interceptor verifies if the user session still active; if there is no active session, the user is redirected to the login page. It currently uses the COPlanS server for the authentication mechanism and to retrieve user properties and preferences. The login package is based on the design pattern “Intercepting filter”. Its description is [10]:

Intercepting filter-- The Intercepting Filter pattern wraps existing application resources with a filter that intercepts the reception of a request and the transmission of a response. An intercepting filter can pre-process or redirect application requests, and can post-process or replace the content of application responses. Intercepting filters can also be stacked one on top of the other to add a chain of separate, declaratively-deployable services to existing Web resources with no changes to source code.

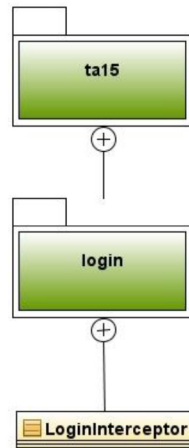



Figure 15 : login package break down

Action Package

This package provides the common interface for implementing all the actions available through the application. It also contains generic actions such as the generation of the initial web page; proceed to the application exit; and intercept changes on the commandment selection, in order to customize the application views.

Table 5 : WelcomeAction user interaction

User interaction		Behavior (Controller)
Login		User is authenticated and redirected to the application welcome page.

After a valid user authentication, the WelcomeAction is called to execute all the functions and retrieve all the data needed for the generation of the welcome.jsp page. Figure 18 shows the dependencies between that action and the business objects beans.

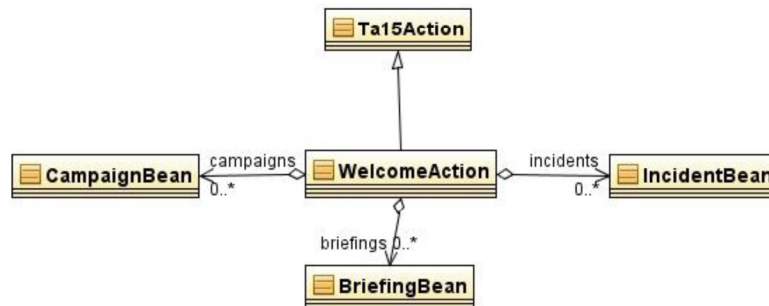


Figure 18 : WelcomeAction dependencies

An example of the associated view generated is illustrated in Figure 19.




Figure 19 : TA15 welcome.jsp

LogoutAction

This action terminates the user session on the application. Being part of the header of the application, it can be executed from any screen page by a simple click on the door icon (Figure 19).

Table 6 : LogoutAction user interaction

User interaction		Behavior (Controller)
Logout		The session will end and the user will be redirected to the login page.

There is no business object impacted by this action as shown on the diagram of Figure 20, it only handles user session.

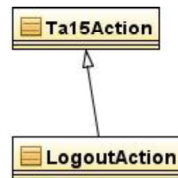



Figure 20 : LogoutAction Dependencies

ChangeCommandmentAction

This action gives the user the possibility of changing the commandment contextual value. That value is used to customize the information of interest to be display and its default value is set by the user organization. In the case of the CANADACOM commandment value, a second field needs to be specified to identify the regional commandment.

Table 7 : ChangeCommandmentAction user interaction

User interaction		Behavior (Controller)
Commandment selection		<ul style="list-style-type: none">• For operations and incidents, only the information part of the area of responsibility of the specified commandment will be displayed.• For the documents and briefings list, an explicit configuration can be set

Regional commandment selection	<div data-bbox="630 195 737 436"> <div data-bbox="630 195 737 247">JTFG</div> <div data-bbox="630 247 737 436"> All JTFG JTFP JTFW JTFC FOI(E) JTFA JTFN </div> </div>	independently for each commandment value.
--------------------------------	---	---

This function is part of the header of the application as illustrated in Figure 19 example.

There is no business object impacted by this action as shown on the diagram of Figure 21, it only handles contextual parameters.



Figure 21 : ChangeCommandmentAction dependencies

Bean Package

This package contains the generic data maintain by the application, it is limited to the application user and its preferences (Figure 22).

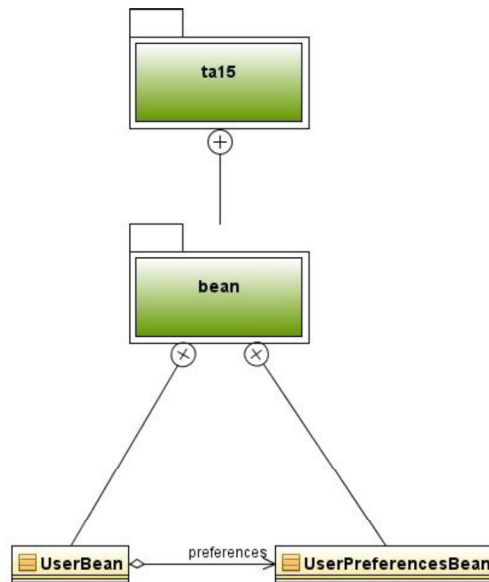


Figure 22 : bean package breakdown

Utils Package

This package contains the generic utility functions for the application (Figure 23). It gives the means to compress all data exchanged with the handheld device and to generate and send email to the user for notification purpose. It also provides utilities to manipulate XML documents and Zulu time format.

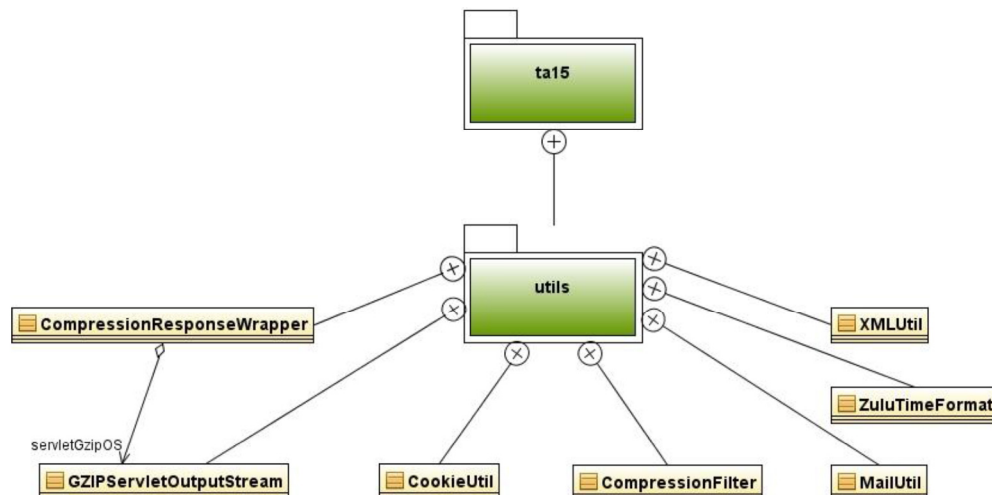


Figure 23 : utils package breakdown

Exception Package

This package contains the functions needed for error management in the application (Figure 24). The information related to the exception is printed on the server console for administration purposes.

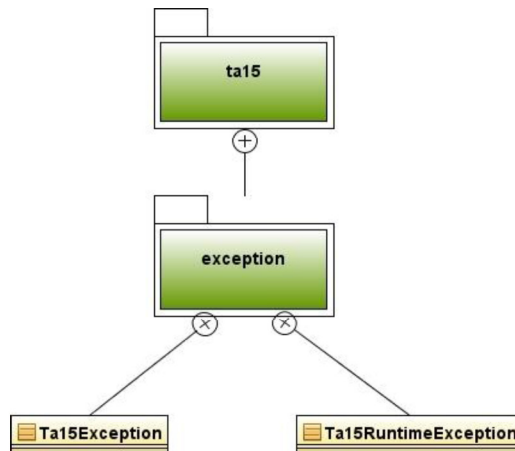


Figure 24 : exception package breakdown

Listener Package

This package contains functions to initiate the application by loading all configurations needed and maintaining users' sessions on the server (Figure 25).

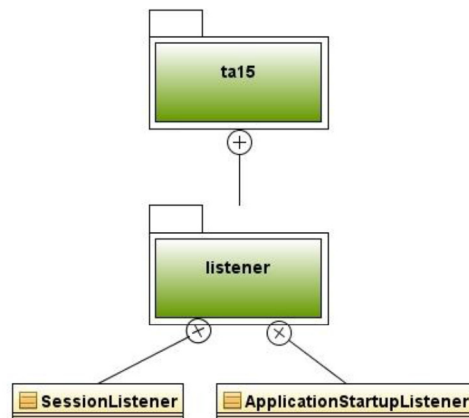
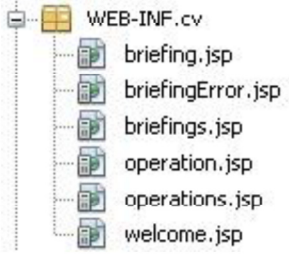


Figure 25 : listener package breakdown

CV Package

This package contains Command View (CV) system functions and related information managed by the application. It is composed of three main packages: bean, actions and utils (Figure 26). The associated Java Server Pages are listed in Table 8.

Table 8 : cv.jsp list

Web Application Structure	JSP	Description
	briefing.jsp	Manage the display; the document name with the related link to the contextual download path to the Content Management System of JCDS 21 Testbed.
	briefingError.jsp	Display error message when a problem occurs on the document access.
	briefings.jsp	Display the briefings list.
	operation.jsp	Display key information about an operation, including its status and related incidents from IMS system.
	operations.jsp	Display the list of operations by category: current, future, past and under planning.
	welcome.jsp	Display the starting page of Command View access module.

The bean package manages information related to the configuration of the document catalogue to be displayed based on the user's organization and level of command. It also contains operations generic attributes managed by Command View.

The action package essentially provides the application logic to access and manipulate the data related to the bean package.

The utils package contains the necessary functions to compress files to be transferred across the network for bandwidth optimization. It uses the NXPowerLite File Server (*REF*) software to reduce Microsoft Office document size.

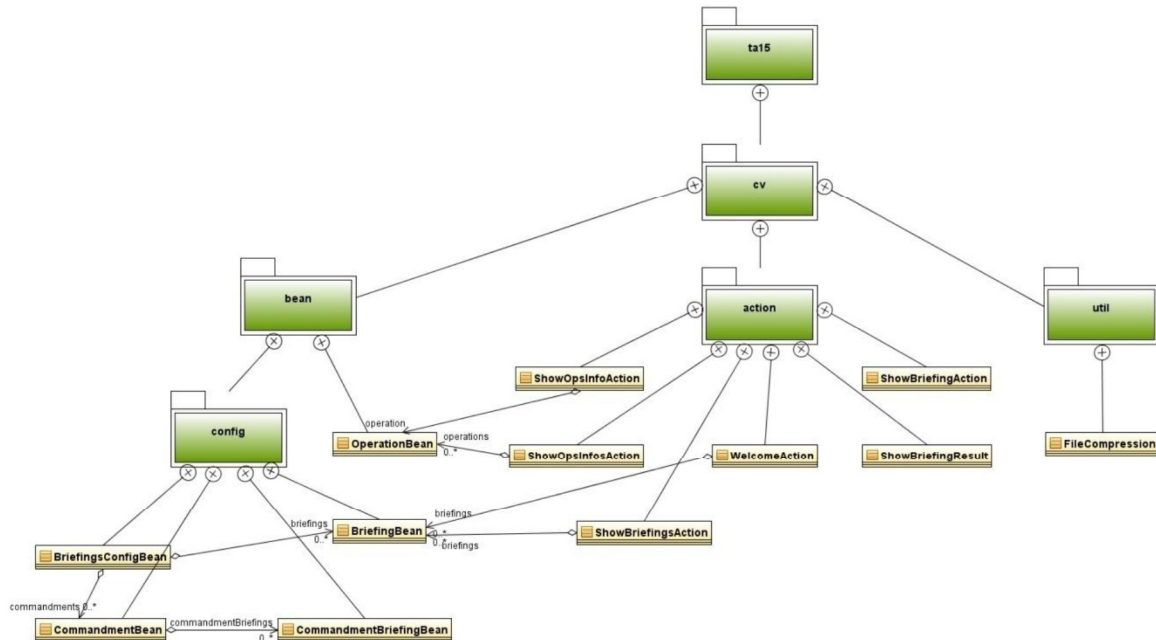


Figure 26 : cv package breakdown

ShowOpsInfosAction and ShowOpsInfoAction

ShowOpsInfosAction and ShowOpsInfoAction actions have been created for test purposes during the development phase while other systems like EMPA and COPlanS were not available. ShowOpsInfosAction creates and manages the list of operations to be displayed by the operations.jsp page, based on the commandment parameters set by the user. This action is mainly called up from the welcome page of the application.

ShowOpsInfoAction provides the possibility of accessing detailed information about an operation. It executes all the functions and retrieves all the data needed for the generation of the operation.jsp page. That action is called from the operations list page when the user clicks on the name of a specific operation.

As shown in Figure 27 and Figure 28, the only dependency of those two actions class is with the OperationBean class.

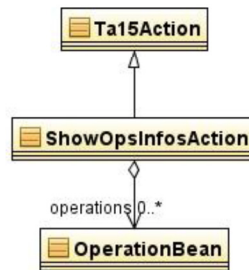


Figure 27 : ShowOpsInfosAction dependencies



Figure 28 : ShowOpsInfoAction dependencies

In the final release of the system, those actions have been replaced by the EmpaPlanExecutionListAction described in section 0 of this document.

ShowBriefingsAction

This action creates and manages the list of important documents and briefings to be displayed by the briefings.jsp page, based on the commandment parameters set by the user. This action is mainly called up from the welcome page of the application. There are two types of document lists that can be customized for each commandment: Strategic Info and Operation Info.

Table 9 : ShowBriefingsAction user interaction

User interaction		Behavior (Controller)
Single click on <u>Strategic Info</u> or <u>Operational Info</u> link	Command View <ul style="list-style-type: none"> • <u>Strategic Info</u> • <u>Operational Info</u> • <u>IMS</u> • <u>Current Ops</u> 	The user will be redirected to the briefings list page for the corresponding level of planning.

As shown in Figure 29, the only dependency of that action class is with the BriefingBean class.

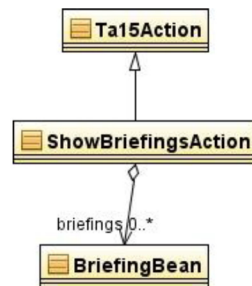


Figure 29 : ShowBriefingsAction dependencies

The page generated by `briefings.jsp` is composed of a single list of documents ordered based on the configuration given by the `BriefingsConfigBean` class. An example of this page is illustrated by the Figure 30.



Figure 30 : cv briefings.jsp

WelcomeAction

This action creates and manages the list of important documents and briefings to be displayed by the `welcome.jsp` page, based on the commandment parameters set by the user. This action is mainly called up from the welcome page of the application. The list is composed of two sections of documents that can be customized for each commandment: Strategic Info and Operation Info. This action is not currently used by the application and is now replaced by the `ShowBriefingsAction`.

Table 10 : WelcomeAction user interaction

User interaction		Behavior (Controller)
Single click on <u>Command View</u> link	<u>Command View</u>	The user will be redirected to the documents list page.

As shown in Figure 31 the only dependency of that action class is with the BriefingBean class.



Figure 31 : WelcomeAction dependencies

The page generated by welcome.jsp is a list composed of two sections of documents selected and ordered based on the configuration given by the BriefingsConfigBean class. An example of this page is illustrated by the Figure 32.

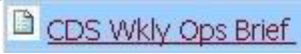


Figure 32 : cv welcome.jsp

ShowBriefingAction

This action allows a document to be retrieved from the Content Management System of the JCDS 21 Testbed. Once retrieved, the web browser offers the capability to open the document with the associated application or to locally download the file.

Table 11 : IMSAction user interaction

User interaction		Behavior (Controller)
Single click the document name link		The file will be retrieved, downloaded and then displayed using the associated tool (e.g. Microsoft Word).

There is no business object impacted by this action as shown on the diagram Figure 33.

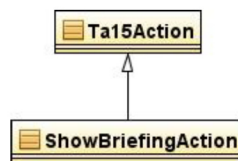


Figure 33 : ShowBriefingAction dependencies

Figure 34 shows an example of opening a briefing using the handheld Microsoft Power Point viewer tool.

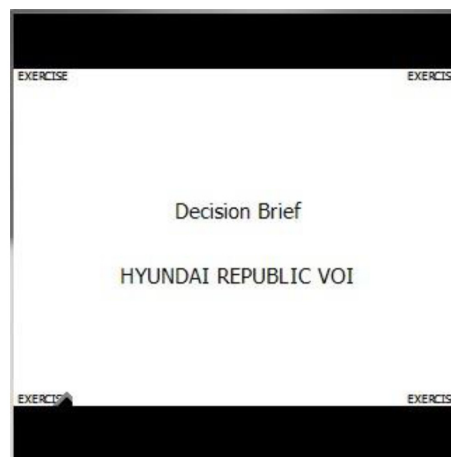


Figure 34 : cv briefing.jsp – display

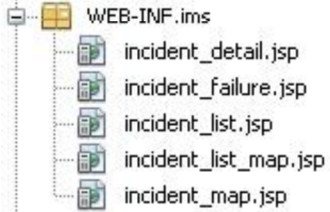
ShowBriefingResult

This action is used internally by the application as a function to handle document transfer through HTTP protocol.

IMS Package

This package contains Incident Management System (IMS) functions and related information managed by the application. It is composed of four major packages: client, action, bean, servlet and map (Figure 35). The associated Java Server Pages are listed in Table 12.

Table 12 : ims jsp list

Web Application Structure	JSP	Description
	incident_detail.jsp	Display the detailed information about an incident.
	incident_failure.jsp	Display an error message when retrieving an incident data.
	incident_list.jsp	Display the sorted incidents list.
	incident_list_map.jsp	Display a map with all the geo locations of the incidents on corresponding list.
	incident_map.jsp	Display a map with the geo location of a particular incident.

The client package includes the functions that allow access to the IMS web service in order to retrieve a filtered incident list based on user parameters.

The bean package maintains detailed information about an incident and keeps the configuration of the commandment area of responsibility, used to filter the incidents list.

The action package essentially provides the application logic to access and manipulate the data related to the bean package.

The servlet and map package provide the functions used to access, generate and manipulate the map images. It sets the map region based on the commandment area of responsibility, filters the incidents list based on this area and generates the map image using the GIS service to get the background layers and superpose incident locations icons.

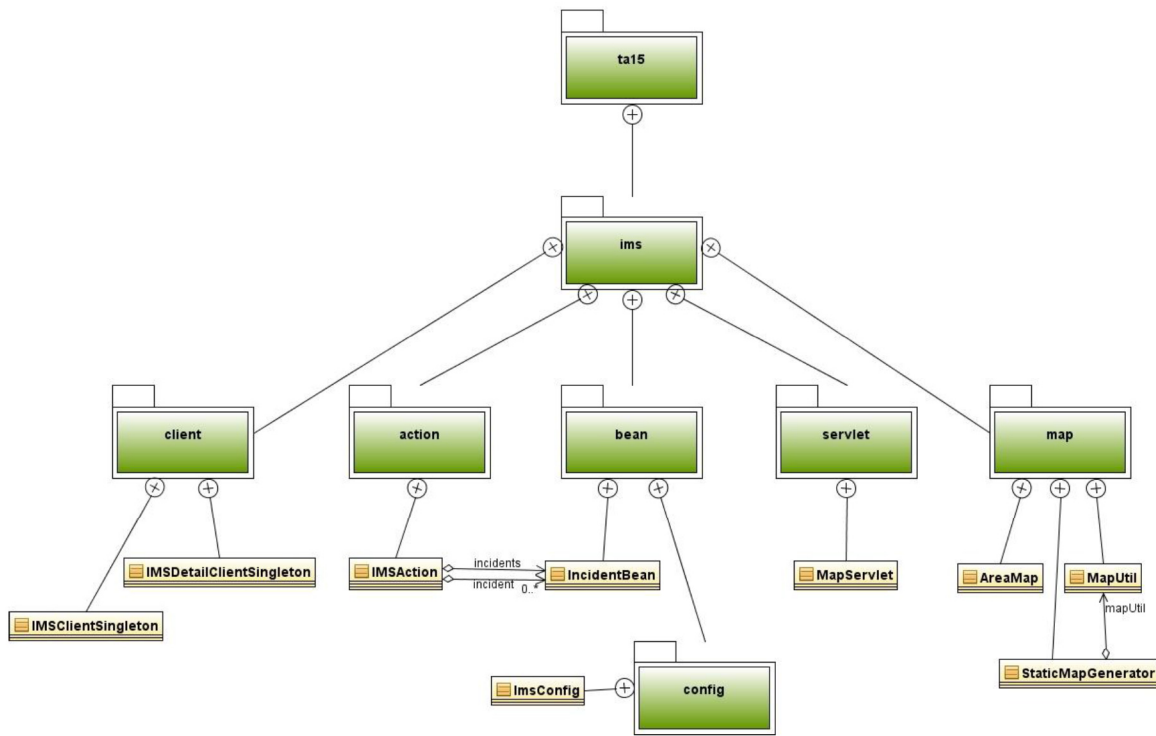


Figure 35 : ims package breakdown

IMSAction

This action groups all the functionalities needed to handle incidents from the Incident Management System (IMS). It can display four different types of pages: a simple list of incidents; a list of incidents geo referenced on a map; detailed information about an incident and the view of a specific incident on the map. This action uses the map package to support the map creation and display.

Table 13 : IMSAction user interaction

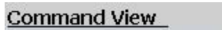


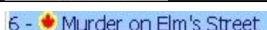

User interaction		Behavior (Controller)
Single click on <u>Command View</u> link		The user will be redirected to the geo referenced list of incidents page.
Single click on <u>IMS</u> link		The user will be redirected to the list of incidents page.
Time filtering selection		Only the incidents that respect the time constraint will be displayed.
Single click on the incident name in the incident list		The user will be redirected to the incident detailed information.
Single click on <u>Map</u> link		The user will be redirected to the incident map page.

Figure 36 shows the dependencies between that action and the business objects beans. It mainly uses the incident bean but also uses EMPA plan information to complete the incident details when the incident is related to the plan.

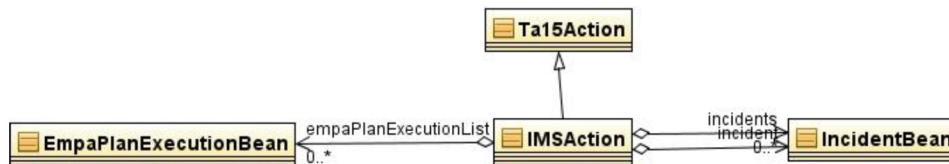


Figure 36 : IMSAction dependencies

The following screen captures illustrate the different displays generated by the associated Java Server Pages. The first example is a simple table list of incidents as shown on Figure 37. The list can be filtered using the combo box to specify time constraint. The three attributes displayed for each incident are Name, Status and Type and the list can be ordered by one of these attributes by clicking on its respective column name.



Name	Status	Type
Test Incident 5 (2008-09-23)	Opened	Incident
Test Incident 2	Opened	Incident
Test Incident 1	Opened	Event

Figure 37 : incident list jsp

The incident details displayed in Figure 38 are a tabular representation of the information available in the incident bean. The map view screen of the incident is available from this screen by clicking on the Map link in the table header.



Future Ops/Assessment :	
Location :	Vancouver
Classification :	Unclassified
Type :	Incident
Category :	Test Category
Operation :	Operation Podium

Figure 38 : incident detail jsp

Figure 39 displays the list of incidents with the corresponding geographic location. As mentioned previously in section 0, the incidents list is always filtered with respect to the commandment selection. Moreover, in this particular screen, the map background depends on that selection; only the area of responsibility of the specific commandment will be displayed.



Figure 39 : incident list map.jsp

While Figure 39 displays a global view of the incidents, Figure 40 is a zoomed view map display of a specific incident.

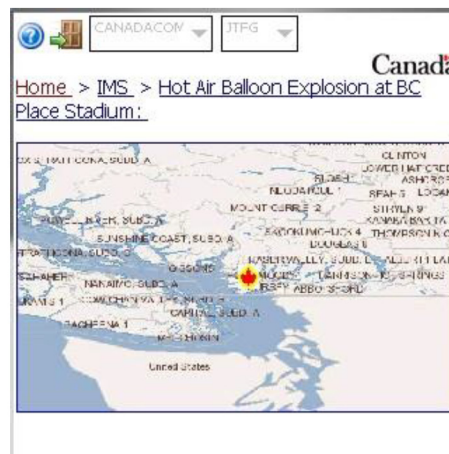
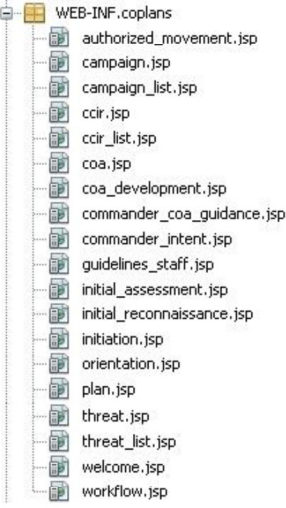


Figure 40 : incident map.jsp

COPlanS Package

This package contains Collaborative Operation Planning System (COPlanS) functions and related information managed by the application. It is composed of four major packages: client, tag, bean and action (Figure 41). The packages bean and action will be detailed in the following sub-sections. The associated Java Server Pages are listed in Table 14.

Table 14 : coplans jsp list

Web Application Structure	JSP	Description
	authorized_movement.jsp	Display the authorized movement form that allows showing, editing and saving the related information.
	campaign.jsp	Display the campaign information and its related plans by their respective level of planning: strategic, operational and tactical.
	campaign_list.jsp	Display the list of campaign managed by COPlanS, including the contingency campaign.
	ccir.jsp	Display the critical commander information requirement (CCIR) form that allows showing, editing and saving the related information
	ccir_list.jsp	Display a table that lists the CCIR related to the plan. That table can be sorted and provides the possibility of creating a new element.
	coa.jsp	Display the properties of a course of action (COA).
	coa_development.jsp	Display the starting page for COA development stage of the planning process. It mainly gives entry point to the available functions for this module.
	commander_coa_guidance.jsp	Display the commander COA guidance form that allows showing, editing and saving the related information.
	commander_intent.jsp	Display the commander intent form that allows showing, editing and saving the related information.
	guidelines_staff.jsp	Display the guidelines to staff form that allows showing, editing and saving the related information.

	initial_assessment.jsp	Display the initial assessment form that allows showing, editing and saving the related information.
	initial_reconnaissance.jsp	Display the initial reconnaissance form that allows showing, editing and saving the related information.
	initiation.jsp	Display the starting page for the initiation stage of the planning process. It mainly gives an entry point to the available functions for this module.
	orientation.jsp	Display the starting page for orientation stage of the planning process. It mainly gives an entry point to the available functions for this module.
	plan.jsp	Display the properties of the plan.
	threat.jsp	Display the properties of a particular threat, including its risk level.
	threat_list.jsp	Display a table that lists the threats related to the plan.
	welcome.jsp	Display the starting page of COPlanS access module.
	workflow.jsp	Display a table of all the stages of the planning process.

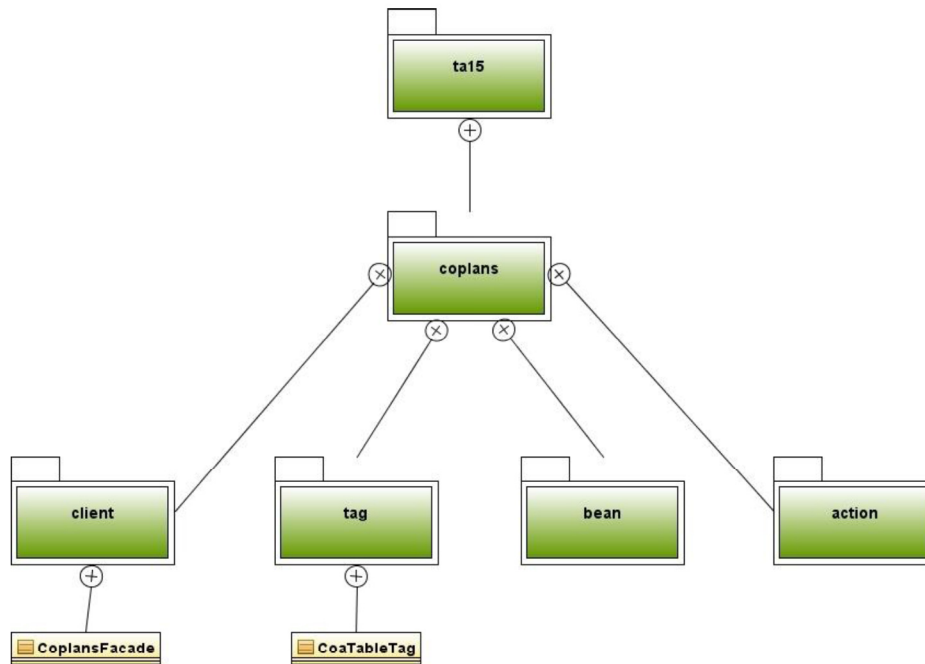


Figure 41 : coplans package breakdown

The client package provides the functionalities that allows access to COPlanS services. Part of those services are the login management of the application that uses COPlanS authentication service and the different functions that allow retrieval and modification of some of the planning information managed by COPlanS system. This package implements the “Session façade” design pattern. Its description is [10]:

***Session facade**--This pattern defines a higher-level business component that contains and centralizes complex interactions between lower-level business components. A Session Facade is implemented as a session enterprise bean. It provides clients with a single interface for the functionality of an application or application subset. It also decouples lower-level business components from one another, making designs more flexible and comprehensible.*

The tag package is fundamentally a utility package that offers formatting functions for more complex data display. It supports the course of actions (COA) table generated by the application.

COPlanS Bean Package

This package provides all the support needed to manipulate COPlanS system data. This includes information on campaign and the related plans, the commander critical information requirements (CCIR), the course of action (COA), the threats and their level of risk, the planning document deliverables and the planning process including its associated steps (Figure 42).

It is important to note that COPlanS beans content, unlike other beans of the application, can also be modified by the user. The resulting change to the data will be reflected in real time to COPlanS

client. One important bean is the PlanBean that contains all the guidelines and the directive from the commander.

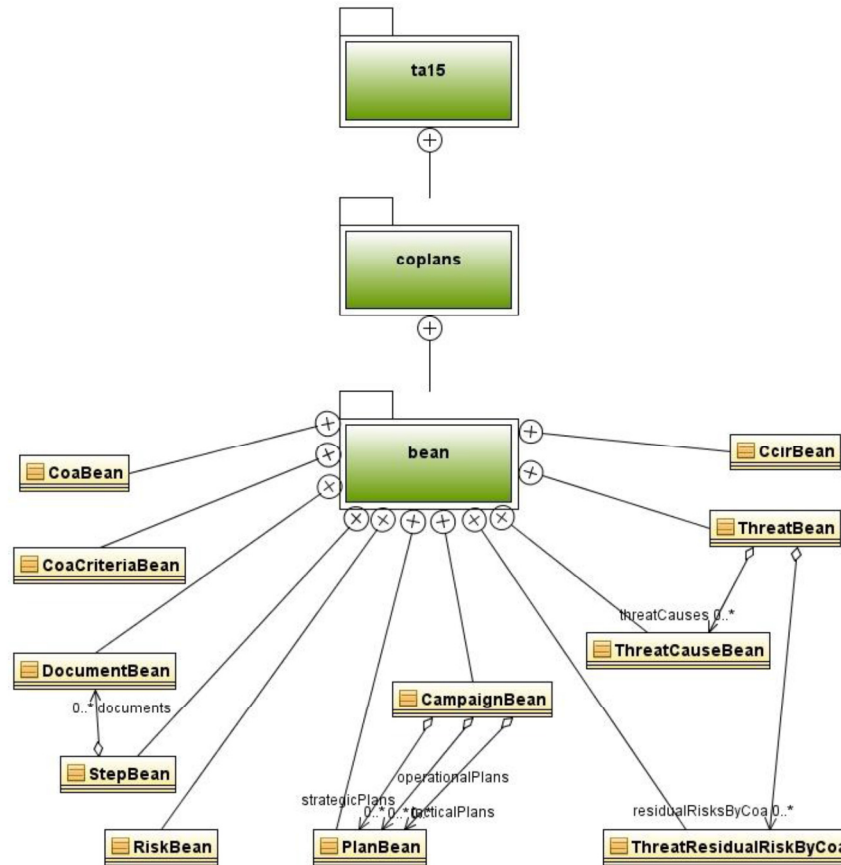


Figure 42 : coplans bean package breakdown

COPlanS Action Package

This package provides the application logic to access and manipulate the data related to the COPlanS bean package (Figure 43). It allows access to the three first steps of the planning process covered by COPlanS, in particular to functions important for the commander.

The dependence between the different actions and the related beans will be explained in the following sections, except for Util and ShowDeliverableResult that are utility classes needed by other COPlanS action classes.

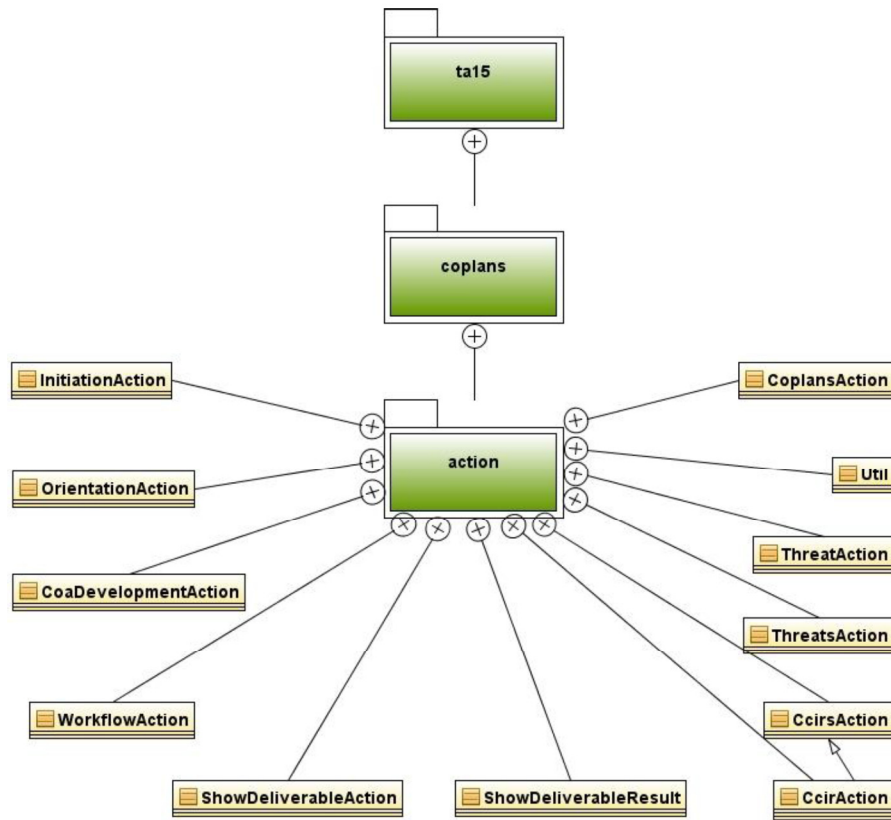


Figure 43 : coplans action package breakdown

CoplansAction

This action is used as the main point of entry in the COPlanS system. It mainly allows retrieval of the different campaigns managed by the system and their associated plans. It allows the user to set the context of the current campaign and plan through the following actions.

Table 15 : CoplansAction user interaction

User interaction		Behavior (Controller)
Single click on <u>COPlanS</u> link	<u>COPlanS</u> <u>OP PODIUM</u> <u>Contingency Plans</u>	The user will be redirected to the complete list of operations.
Single click on the campaign name link or on the “Contingency Plans” link		The user will be redirected to the list of plans related to the campaign or to the contingency plans. The context of the application will be set to that campaign for the following actions.
Single click on the plan name link	Campaign OP PODIUM Strategic Plans : <ul style="list-style-type: none">• <u>OP PODIUM (Strat.Lvl)</u> Operational Plans : <ul style="list-style-type: none">• <u>OP PODIUM (Op.Lvl)</u>	The user will be redirected to the descriptive properties of the plan and the context of the application will be set to that plan for the following actions.

Figure 44 shows the dependencies between that action and the business objects beans. The CampaignBean manages information about the past, current and future campaigns and maintains the contingency plans through a special “Contingency Plan” campaign. This dual function explains the two dependencies with the action on the diagram.

The PlanBean manages all the information about the plan but in the particular used of this action, it retrieves the descriptive properties to be displayed.

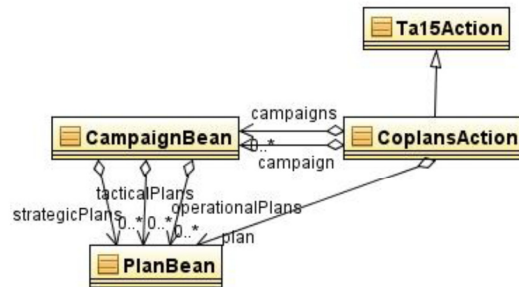


Figure 44 : coplans action dependencies

Figure 45 shows the campaigns list page of COPlanS.

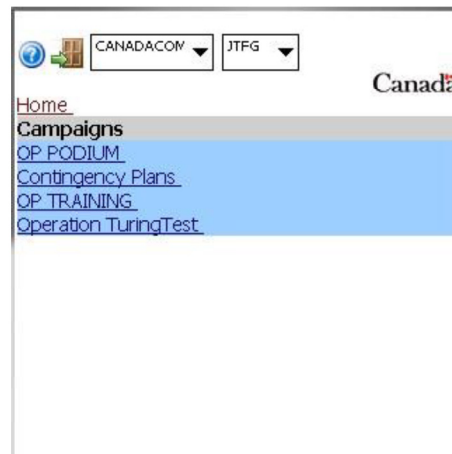


Figure 45 : coplans campaign list.jsp

Figure 46 illustrates an example of a list of plans related to a specific campaign. The plans are grouped into three categories: Strategic Plans, Operational Plans and Tactical Plans.



Figure 46 : coplans campaign.jsp

Figure 47 is an example of the plan properties page. This is also the starting page to all other main actions available through COPlanS part of the application.



Figure 47 : coplans plan.jsp

InitiationAction

This action permits interaction with three important tasks in the first stage of the planning process managed by COPlanS: the authorized movement, the guidelines to staff and the initial reconnaissance. For those three tasks, this action retrieves the associated information to be displayed and provides the update mechanism that allows this information to be modified from the CHESS application.

Table 16 : InitiationAction user interaction

User interaction		Behavior (Controller)
Single click on <u>Authorized Movement</u> , <u>Guidelines To Staff</u> or <u>Initial Reconnaissance</u> link	Initiation <u>Authorized Movement</u> <i>Freedom of movement in conjunction with the Coast Guard, RCMP</i> <u>Guidelines To Staff</u> <i>1. SITUATION. a. General. This...</i> <u>Initial Reconnaissance</u> <i>See OP PODIUM Op Plan</i>	The user will be redirected to the corresponding page to consult or modify the information.

Figure 48 shows the dependencies between that action and the business objects beans. The StepBean manages the information related to the deliverable documents while the PlanBean contains the information of the authorized movement, guidelines to staff and initial reconnaissance.

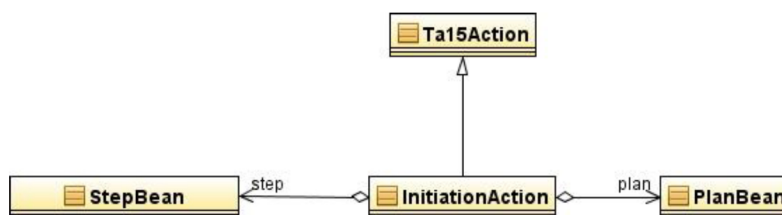


Figure 48 : InitiationAction dependencies

Figure 49, Figure 50 and Figure 51 illustrate the three screens generated to visualize and modify the authorized movement, guidelines to staff and initial reconnaissance. The user can scroll the text box to consult the information, type directly in the text box to modify the information and click the “Save” button to apply its changes.



Figure 49 : coplans authorized movement.jsp

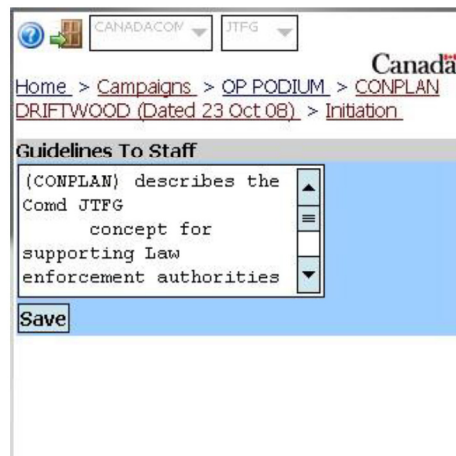


Figure 50 : coplans guidelines staff.jsp



Figure 51 : coplans initial reconnaissance.jsp

OrientationAction

This action allows interaction with two of the commander's important tasks in the second stage of the planning process managed by COPlanS: the commander's intent and the commander's COA guidance. For those two tasks, this action retrieves the associated information to be displayed and provides the update mechanism that allows information to be modified from the CHES application.

Table 17 : OrientationAction user interaction

User interaction		Behavior (Controller)
Single click on <u>Commander's Intent</u> or <u>Commander's COA</u> <u>Guidance</u> link	Orientation Commander's Intent <i>GENERAL (a) The handling of a V...</i> Commander's COA Guidance <i>Scope The development of this p...</i>	The user will be redirected to the corresponding page to consult or modify the information.

Figure 48 shows the dependencies between that action and the business objects beans. The StepBean manages the information related to the deliverable documents while the PlanBean contains the information of the commander's intent and commander's COA guidance.

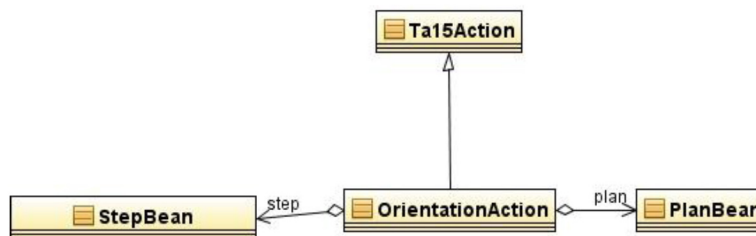


Figure 52 : OrientationAction dependencies

Figure 53 and Figure 54 illustrate the three screens generated to visualize and modify the authorized movement, guidelines to staff and initial reconnaissance. The user can scroll the text box to consult the information, type directly in the text box to modify the information and click the “Save” button to apply its changes.

Figure 53 : coplans commander intent.jsp

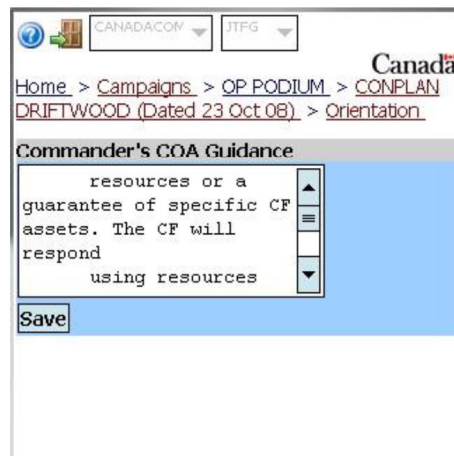


Figure 54 : coplans commander coa guidance jsp

CoaDevelopmentAction

This action handles information related to the COA (Course Of Action) developed through the COPlanS system. It retrieves important information used by the commander to decide which options will be used to develop the plan. This action automatically transmits the commander's decision to the COPlanS planning team.

Table 18 : CoaDevelopmentAction user interaction

User interaction		Behavior (Controller)
Single click on COA name link		The user will be redirected to the COA properties and description page.
Single click on the COA radio box		The corresponding COA will be selected.

Figure 55 shows the dependencies between that action and the business objects beans. The StepBean manages the information related to the deliverable documents while the CoaBean contains description and evaluation information about the COA. One of the most important uses of PlanBean is to allow retrieval and setting of the selected COA for the plan.

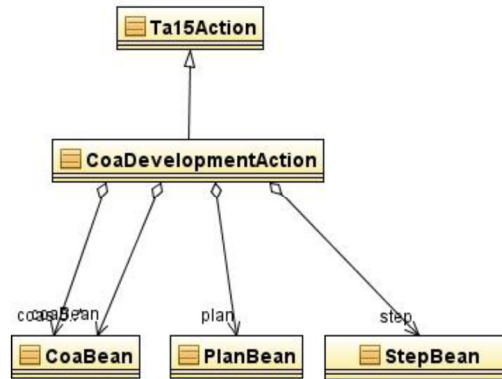


Figure 55 : CoaDevelopmentAction dependencies

Figure 56 shows an example of a table that allows COA evaluation information to be displayed as well as the selection of the COA to be used for plan development.

Government of Canada / Gouvernement du Canada

Canada

Help Logout CANADACOM UTF8

Home > Campaigns > Campaign Test 1 > Plan Test 1 > Plan Workflow

COAs		COA Test	COA Test A
Ranking		Development In Progress	
Selected		<input type="radio"/>	<input type="radio"/>
Criteria	Weight		
Covering Operational Tasks	0.1	1	1
Covering Mission's Possible Locations	0.05	1	0.5
Covering Enemy's Courses of Actions	0.05	0.5	0.5

Figure 56 : coplans coa development jsp (1)

Figure 57 is a simple properties information display page for a specific COA.

The screenshot shows a web application interface for the Government of Canada. At the top, there are logos for the Government of Canada and the Canadian flag, along with the text 'Canada'. Below this is a navigation bar with 'Help' and 'Logout' links, and dropdown menus for 'CANADACOM' and 'JTFC'. A breadcrumb trail reads: 'Home > Campaigns > Campaign Test 1 > Plan Test 1 > Plan Workflow > COAs'. The main content area is titled 'COA' and displays the following properties for 'COA Test':

Name :	COA Test
ID :	C1
Earliest Starting Date :	082103Z Sep 2008
Latest Ending Date :	082103Z Sep 2008
Status :	Development In Progress
Team Leader :	admin
Affiliation :	Own
Complete Description :	
Human Reliability :	
Remarks :	Remarks
Planning Officers :	

Figure 57 : *coplans coa.jsp*

WorkflowAction

This action is used to retrieve information for all steps of the workflow associated with the planning process. This information mainly gives the status of the execution of the workflow. The commander has the possibility of controlling the battle rhythm of the process by starting or ending a specific step.

Table 19 : *WorkflowAction user interaction*



User interaction		Behavior (Controller)
Single click on <u>Start</u> link		Corresponding step of the workflow will be activated and contributor will be notified.
Single click on <u>Stop</u> link		Corresponding step of the workflow will terminate.

Figure 58 shows the dependencies between that action and the business objects beans. Information associated with the workflow step is managed by the StepBean except the starting step that is managed by the PlanBean.

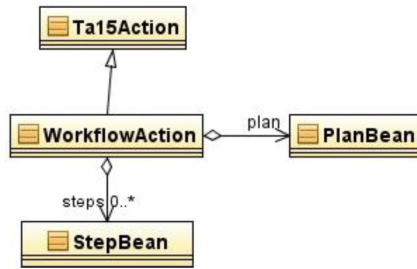


Figure 58 : WorkflowAction dependencies

An example of the page generated by the workflow.jsp is illustrated in Figure 59. The planning process activity associated with the step can be directly accessed by clicking on the activity name link (e.g.: [Initiation](#)).

Plan	Status	Start Date	End Date
		9/29/08 4:36 PM	9/29/08 4:36 PM
Initiation	Completed	planned	planned
		9/29/08 4:36 PM	9/29/08 4:36 PM
		9/29/08 4:36 PM	End
Orientation	Processing	planned	planned
		:	9/29/08

Figure 59 : coplans workflow.jsp

Threats and ThreatAction

Threats are stored in a list array and ThreatsAction is responsible for maintaining that list including adding and removing a threat. On the other hand, ThreatAction manages a specific threat element including changing some of its properties.

Table 20 : ThreatsAction and ThreatAction user interaction



User interaction		Behavior (Controller)
Single click on <u>TRA</u> (Threat And Risk Assessment) link	 <u>TRA</u>	The user will be redirected to the threats list of the actual plan.
Single click on threat name link	 <u>Assymetric (VOI) Threat</u>	The user will be redirected to the threat properties and description page.

Figure 60 shows the dependencies between those two actions and the business objects beans. The PlanBean is needed to retrieve whichever plan is attached to the threats list and the ThreatBean is the container that will keep all the properties associated with a threat.

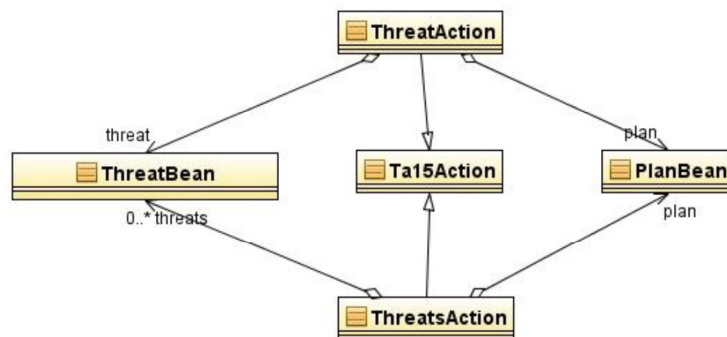


Figure 60 : ThreatsAction and ThreatAction dependencies

Figure 61 illustrates an example of a threat list. On the left side of each element, a small square is generated and filled with the color of the risk evaluation value. The color corresponds to the doctrine evaluation matrix (REF). It should be noted that the current interfaces do not allow the addition or removal of a threat from the list even if the ThreatsAction provides this capability.



Figure 61 : coplans threat list.jsp

Figure 62 provides an example of a threat properties display. It should be noted that the current interfaces do not allow editing of the threat properties even though the ThreatAction provides t capability.



Figure 62 : coplans threat.jsp

CCIRs and CCIRsAction

CCIRs are stored in a list array and CcirsAction is responsible for maintaining that list including adding and removing a CCIR. That action also allows the list to be sorted by a choice of three properties: Name, Status and Importance. On the other hand, ThreatAction manages a specific threat element including changing and updating its properties.

Table 21 : CCIRs and CCIRsAction user interaction


User interaction		Behavior (Controller)
Single click on <u>CCIRs</u> link	 CCIRs	The user will be redirected to the CCIRs list related to the actual plan.
Single click <u>New</u> link	CCIRs New	The user will be redirected to a new CCIR form that will allow the filling and editing of all the required properties.
Single click on <u>Name</u> , <u>Importance</u> and <u>Closed</u>	Name Importance Closed	CCIRs list will be sorted by Name, Importance or Status (closed or not)
Single click on CCIR name link	New Comd's Initial Operational Information Requirement	The user will be redirected to the CCIR properties and description page.

Figure 60 shows the dependencies between those two actions and the business objects beans. The PlanBean is needed to retrieve whichever plan is attached to the CCIR list and the CCIRBean is the container that will keep all the properties associated to a CCIR.

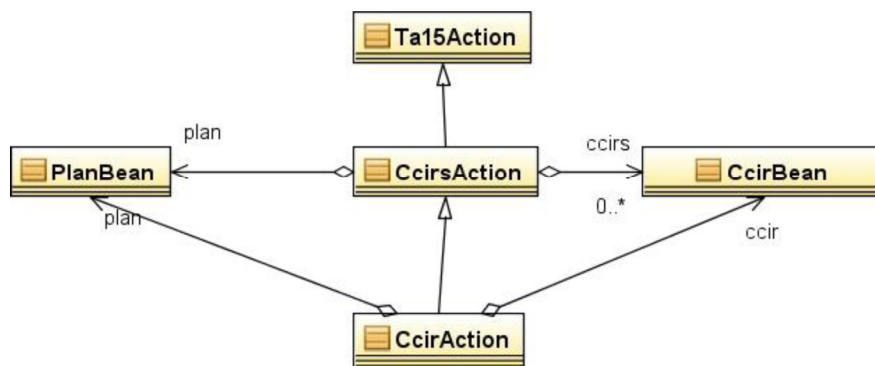


Figure 63 CCIRAction and CCIRAction dependencies

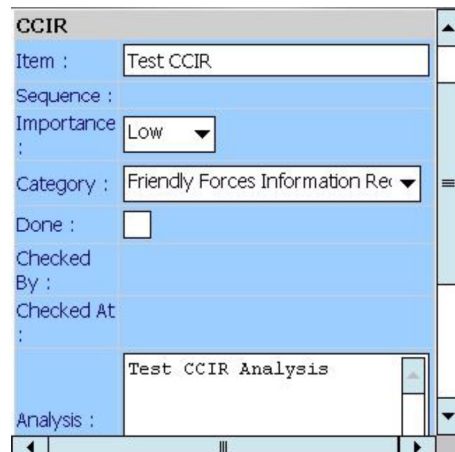
Figure 64 displays an example of the table listing the CCIRs produced by the ccir_list.jsp.



Name	Importance	Closed
ccir	Low	<input type="checkbox"/>
New Comd's Initial Operational Information Requirement 1	High	<input checked="" type="checkbox"/>
New Comd's Initial Operational Information Requirement	Normal	<input type="checkbox"/>

Figure 64 : coplans ccir list.jsp

Figure 65 presents the form available to edit or create a new CCIR. A “Save” button at the bottom of the form (not shown on the picture) applies the changes made by the user.



CCIR	
Item :	Test CCIR
Sequence :	
Importance :	Low
Category :	Friendly Forces Information Rex
Done :	<input type="checkbox"/>
Checked By :	
Checked At :	
Analysis :	Test CCIR Analysis

Figure 65 : coplans ccir.jsp

ShowDeliverableAction

This action gives the user the possibility of retrieving deliverable documents from the planning process activities. The document can be opened and consulted from the wireless device.

Table 22 : ShowDeliverableAction and ShowDeliverableResult user interaction

User interaction		Behavior (Controller)
Single click on document name	Deliverables WarningOrder.doc 188 KB 10/21/08 9:15 AM MissionAnalysisBrief.ppt 290 KB 10/21/08 11h07 AM	The document will be downloaded from the server to the wireless device and then display using the associated tool (e.g. Microsoft Word).

As shown on Figure 66, ShowDeliverableAction has no dependencies on business object bean.

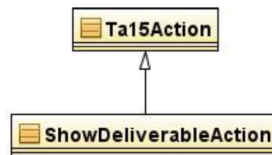


Figure 66 : ShowDeliverableAction dependencies

The lower part of Figure 67, Figure 68 and Figure 69 show examples of deliverables managed by this action for Initiation, Orientation and COA Development activities.



Figure 67 : coplans initiation jsp

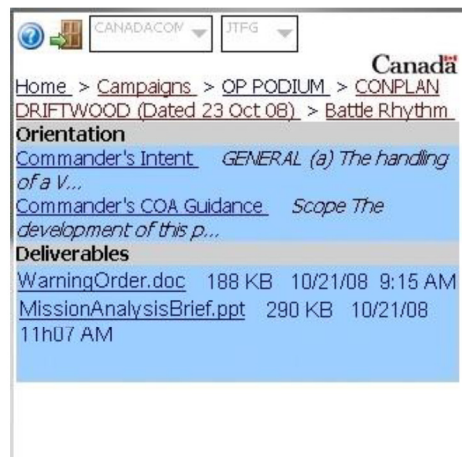


Figure 68 : coplans orientation jsp


Government of Canada / Gouvernement du Canada



[Help](#)
[Logout](#)

CANADACOM
JTFG

[Home](#) >
[Campaigns](#) >
[Campaign Test 1](#) >
[Plan Test 1](#) >
[Plan Workflow](#)

COAs

Ranking

Selected

Criteria

Weight

COA Test

COA Test A

Development In Progress

Covering Operational Tasks

0.1

1

1

Covering Mission's Possible

0.05

1

0.5

Deliverables

InfoBrief.ppt

309.76 KB

10/23/08 7:14 AM

DecisionBrief.ppt

248.832 KB


10/23/08 7:08 AM

Figure 69 : coplans coa development.jsp (2)

EMPA Package

This package contains Execution Management and Plan Adaptation (EMPA) functions and related information managed by the application. It is composed of three major packages: action, bean and client (Figure 70). The associated Java Server Pages are listed in Table 23.

Table 23 : empa.jsp list

Web Application Structure	JSP	Description
 WEB-INF.empa empaplanexec.jsp empaplanexeclist.jsp	empaplanexec.jsp	Display the execution information of a particular plan.
	empaplanexeclist.jsp	Display the list of plans managed by EMPA.

The client package includes the functions that allow accessing the EMPA web service in order to retrieve the list of plans in execution.

The bean package maintains the information about the plan and its execution timing and status.

The action package essentially provides the application logic to access and manipulate the data related to the bean package.

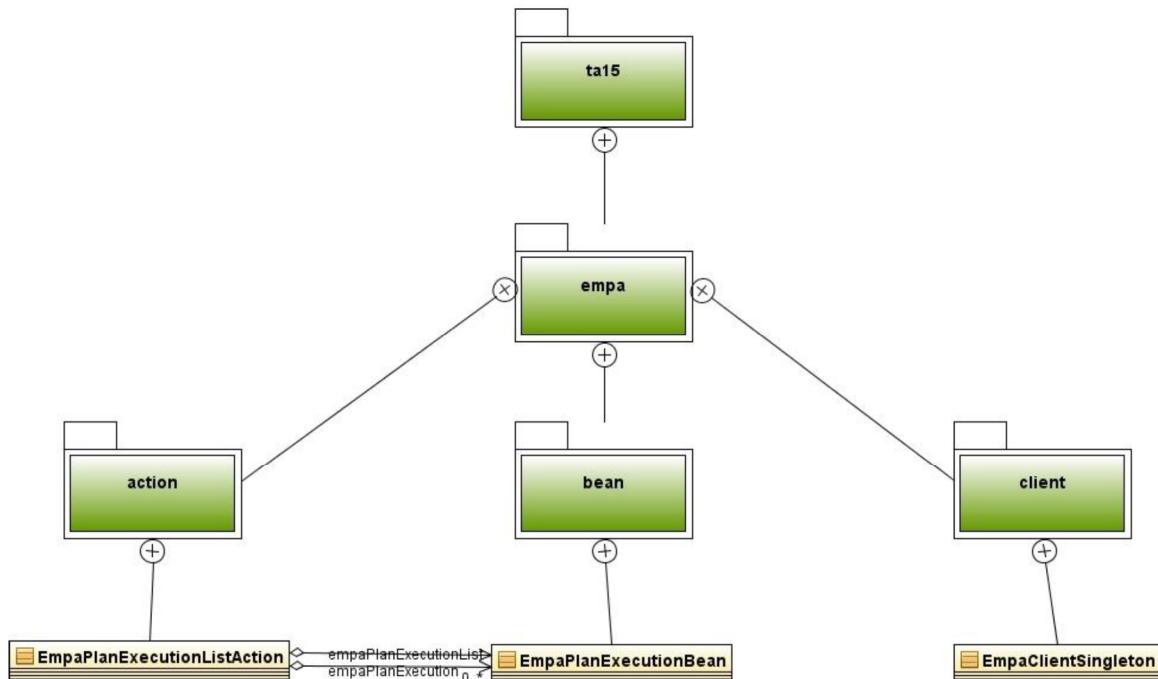


Figure 70 : empa package breakdown

EmpaPlanExecutionListAction

This action creates and manages the list of operations to be displayed by the empaplanexeclist.jsp page based on the commandment parameters set by the user. This action is mainly called up from the welcome page of the application under the Command View section. That action is also called from the operations list page when the user clicks on the name of a specific operation. It executes all the functions and retrieves all the data needed for the generation of the empaplanexec.jsp page.

Table 24 : EmpaPlanExecutionListAction user interaction

User interaction		Behavior (Controller)				
Single click on <u>Current Ops</u> link	<div>Command View</div> <ul style="list-style-type: none">• Strategic Info• Operational Info• IMS• Current Ops	The user will be redirected to the operations list page.				
Single click on operation name link	<div>Current Ops</div> <table><thead><tr><th>Name</th><th>Status</th></tr></thead><tbody><tr><td>Vancouver-Whistler (Improved version)</td><td>Cancelled</td></tr></tbody></table>	Name	Status	Vancouver-Whistler (Improved version)	Cancelled	The user will be redirected to the operation detail page.
Name	Status					
Vancouver-Whistler (Improved version)	Cancelled					

As shown in Figure 71 the only dependency of that action class is with the EmpaPlanExecutionBean class.

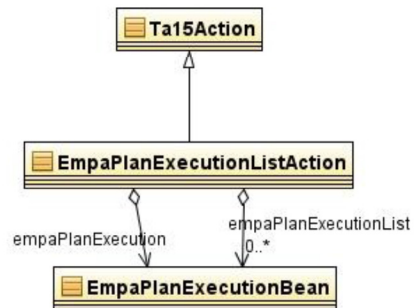


Figure 71 : *EmpaPlanExecutionListAction* dependencies

Only a limited part of the information provided by the OperationBean is used for the list display: its name and status as illustrated in the Figure 72.



Figure 72 : *empa empaplanexeclist.jsp*

Figure 73 presents an example of the operation detailed display. It is mainly a tabular display of the EmpaPlanExecutionBean properties.


 CANADACOM All Canada	
Home > Current Ops	
Vancouver-Whistler (Improved version)	Plan Details
Start Time :	Thu Oct 02 13:48:00 GMT-05:00 2008
Area :	
State :	Cancelled
Progress :	On schedule
Expected Duration Time :	660000 seconds
Running Time :	2 seconds

Figure 73 : *empa empaplanexec.jsp*

4 Conclusion and Future work

CHESS has been developed to demonstrate how a wireless handheld device can be used to support commanders and senior staff officers on the move or with limited connectivity by providing the following capacities:

- Access to situational awareness and planning information;
- Receive real-time notification on their hand held device;
- Collaborate with their staff and provide online guidance and directives;
- Receive briefings;
- Make decisions and approve documents.

This document provides an overview of the high-level architecture that has been used for the implementation of CHESS. This implementation of CHESS can be used on a secure mobile wireless device, a factor which had a direct impact on the way functionalities were designed since we had to use an operations system that was accredited and it was not possible to keep information in memory. However, considering the rapid evolution of mobile wireless handheld technology, it is anticipated that future users' interfaces on such secure devices will be a lot more efficient in terms of interaction and visualization possibilities.

References

- [1] Bélanger, M., Pageau, N. (2010). CHESS: Commander HandhEld Support System, UNCLASSIFIED, 15th International Command and Control Research and Technology Symposium (ICCRTS), Santa Monica CA, 22-24 June 2010.
- [2] Fortin, S. (2009), Commander Wireless Handheld Prototype System Development - TA15 System Requirements Specifications, DRDC Valcartier CR 2009-116, April 2009.
- [3] Fortin, S. (2009), Commander Wireless Handheld Prototype System Development - System Architecture & Requirements Allocation Description (SARAD), DRDC Valcartier CR 2009-117, April 2009.
- [4] Fortin, S. (2008), Identification of Requirements related to the use of a Wireless Handheld Tool for a Commander, Unclassified, CR 2009-119, January 2008.
- [5] CAE PS Canada (2008), JCDS21: Human Factors Style Guide, DRDC Toronto CR, April 2008.
- [6] FUJITSU Consulting (Canada) (2007), JCDS 21 High-Level Functional Architecture, DRDC Valcartier, June 2007.
- [7] FUJITSU Consulting (Canada) (2009), JCDS 21 CTB Software Architecture Description (SAD), DRDC Valcartier CR, January 2009 .
- [8] Chief Information Officer Branch (2007), Common Look and Feel for the Internet 2.0 Standards, Treasury Board of Canada Secretariat, January 2007.
- [9] Raz, M. (2008), JCDS21 Exercise Communications/Network Plan TA 25 Draft V0.5 19 Sep 08, Prolity Corporation, September 2008.
- [10] Alur, D., Crupi, J., and Malks, D., Core J2EE Patterns: Best Practices and Design Strategies, Prentice Hall, 2001.

Annex A Libraries list

Table 25 provides a list of libraries used and needed by the CHESS system. The dependencies between those libraries are not covered by this table. Those libraries are all free open source library or proprietary of DRDC – Valcartier projects.

Table 25 : List of CHESS libraries

Library name	Description
db	Allows accessing COPlanS data persistence layer.
model	Provides COPlanS model implementation.
client	Provides client communication layer to access COPlanS server.
core	This library contains all the core component of COPlanS needed by other COPlanS libraries.
htmlparser	HTML Parser is a Java library used to parse HTML in either a linear or nested fashion. Primarily used for transformation or extraction, it features filters, visitors, custom tags and easy to use JavaBeans.
commons-lang-2.4	The <i>Lang</i> Component provides a host of helper utilities for the java.lang API, notably String manipulation methods, basic numerical methods, object reflection, creation and serialization, and System properties. Additionally it contains an inheritable enum type, an exception structure that supports multiple types of nested-Exceptions, basic enhancements to java.util.Date and a series of utilities dedicated to help with building methods, such as hashCode, toString and equals.
activation	With the JavaBeans Activation Framework standard extension, developers who use Java technology can take advantage of standard services to determine the type of an arbitrary piece of data, encapsulate access to it, discover the operations available on it, and to instantiate the appropriate bean to perform said operation(s). For example, if a browser obtained a JPEG image, this framework would enable the browser to identify that stream of data as a JPEG image, and from that type, the browser could locate and instantiate an object that could manipulate, or view that image.
commons-io-1.4	Commons IO is a library of utilities to assist with developing IO functionality. There are four main areas included: <ul style="list-style-type: none">• Utility classes - with static methods to perform common tasks

	<ul style="list-style-type: none"> • Filters - various implementations of file filters • Comparators - various implementations of java.util.Comparator for files • Streams - useful stream, reader and writer implementations
commons-logging-1.0.4	The Logging package is an ultra-thin bridge between different logging implementations. A library that uses the commons-logging API can be used with any logging implementation at runtime. Commons-logging comes with support for a number of popular logging implementations, and writing adapters for others is a reasonably simple task.
comparaison	The comparaison library is used by COPlanS system for Course of Action (COA) ranking purpose.
FastInfoset	Allows manipulation of Fast Infoset documents. Fast Infoset is an international standard that specifies a binary encoding format for the XML Information Set (XML Infoset) as an alternative to the XML document format. It aims to provide more efficient serialization than the text-based XML format.
freemarker-2.3.8	FreeMarker is a “template engine”; a generic tool to generate text output (anything from HTML to autogenerated source code) based on templates.
htmllexer	Provides low level access to generic string, remark and tag nodes on the page in a linear, flat, sequential manner.
http	Provides support to IITTP protocol
jaxb	JAXB is an acronym derived from Java Architecture for XML Binding. It constitutes a convenient framework for processing XML documents, providing significant benefits as compared to previously available methods such as the one following the Document Object Model (DOM).
jaxws	The Java API for XML Web Services (JAX-WS) is a Java programming language API for creating web services.
JCDSEvent	It allows handling of events from the JCDS.
jsr173_api	The Streaming API for XML (StAX) is a Java based API for pull-parsing XML.
jsr181-api	JSR 181 enables creation of portable Java Web Services from a simple Plain Old Java Object (POJO) class by adding annotations.
jsr250-api	This JSR provides annotations for common semantic concepts in the J2SE and J2EE platforms that apply across a variety of individual technologies.

jstl-1.2	The JavaServer Pages Standard Tag Library (JSTL) encapsulates, as simple tags, core functionality common to many JSP applications.
log4j-1.2.15	With log4j it is possible to enable logging at runtime without modifying the application binary. The log4j package is designed so that these statements can remain in shipped code without incurring a heavy performance cost.
mail	The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications.
Notifiable 2008-06-16	This library allows access to JCDS notification service.
notificationESBWebServiceClient 2008-07-18	This library allows management of notifications from the JCDS notification service.
contentManagementWSClient 2008-05-22D	This library allows access to the content management system service of JCDS.
ognl-2.6.11	OGNL stands for Object-Graph Navigation Language; it is an expression language for getting and setting properties of Java objects.
resolver	The Apache XML Commons Resolver classes implement Catalog-based entity and URI resolution.
saaj	The SOAP with Attachments API for Java™ (SAAJ) 1.2.1 provides the API for creating and sending SOAP messages by means of the javax.xml.soap package. It is used for the SOAP messaging that goes on behind the scenes in JAX-RPC and JAXR implementations.
sax2	SAX is the <i>Simple API for XML</i> ,
sjsxp	The Sun Java Streaming XML Parser (SJSXP) is an efficient implementation of the StAX API which is fully compliant with the XML 1.0 and Namespace 1.0 specifications.
streambuffer	XML Stream Buffer Implementation
struts2-core-2.0.11.1 and xwork-2.0.4	Struts2 is an extensible MVC framework for creating enterprise-ready Java web applications struts2-core and xwork contain the core components and resources of the Struts2 framework.
thumbelina	Used for HTML parsing
xpp3_min-1.1.4c	MXP1 is a stable XmlPull parsing engine that is based on ideas from XPP and in particular XPP2 but completely revised and rewritten to take the best advantage of latest JIT JVMs such as Hotspot in JDK 1.4+.
xstream-1.3	XStream is a unique open-source Java library for serializing objects into XML and deserializing that XML into objects.
junit-3.8.2 and junit-4.5	JUnit is a simple framework for writing and running automated tests.

This page intentionally left blank.

List of symbols/abbreviations/acronyms/initialisms

[Enter list here, if applicable. If not, delete the page.]

API	Application Programming Interface
BPEL	Business Process Execution Language
C2	Command & Control
CANADACOM	Canada Command
CCIR	Commander Critical Information Requirements
CHESS	Commander HandhEld SyStem
CLF	Common Look and Feel
COA	Course of Action
COPlanS	Collaborative Operations Planning System
CSNI	Consolidated Secret Network Infrastructure
CTB	Core Test Bed
CV	Command View
DJFC	Director Joint Force Capability
DOM	Document Object Model
DND	Department of National Defence
DRDC	Defence Research & Development Canada
DRDKIM	Director Research and Development Knowledge and Information Management
DWAN	Departmental Wide Area Network
EMPA	Execution Management and Plan Adaptation
ESB	Enterprise Service Bus
GIS	Geographic Information System
HAIPE	High Assurance Internet Protocol Encryption
HLFA	High Level Functional Architecture
HQ	Head Quate
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol

IMS	Incident Management System
IO	Input/Output
JCDS21	Joint Command Decision Support system for the 21st century
JPEG	Joint Photographic Experts Group
JSP	Java Server Pages
JSTL	JavaServer Pages Standard Tag Library
MVC	Model View Controller
OPP	Operational Planning Process
PDA	Personal Digital Assistant
QVGA	Quarter Video Graphics Array
R&D	Research & Development
SAD	System Architecture Description
SME PED	Secure Mobile Environment Portable Electronic Device
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
TA	Task Authorization
TB	Test Bed
TD	Technology Demonstration
TDP	Technology Demonstration Project
TRA	Threat and Risk Assessment
UDDI	Universal Description, Discovery and Integration
XML	Extensible Markup Language

DOCUMENT CONTROL DATA		
(Security markings for the title, abstract and indexing annotation must be entered when the document is Classified or Designated)		
1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence Research and Development Canada – Valcartier 2459 Pie-XI Blvd North Quebec (Quebec) G3J 1X5 Canada		2a. SECURITY MARKING (Overall security marking of the document including special supplemental markings if applicable.) UNCLASSIFIED
		2b. CONTROLLED GOODS (NON-CONTROLLED GOODS) DMC A REVIEW: GCEC APRIL 2011
3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.) System Design - Commander HandhEld SyStem (CHESS)		
4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used) M. Belanger; N. Pageau		
5. DATE OF PUBLICATION (Month and year of publication of document.) May 2012	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.) 92	6b. NO. OF REFS (Total cited in document.) 10
7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.) Technical Note		
8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.) Defence Research and Development Canada – Valcartier 2459 Pie-XI Blvd North Quebec (Quebec) G3J 1X5 Canada		
9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)	
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC Valcartier TN 2013-303	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)	
11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.) Unlimited		
12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.) Unlimited		

13. **ABSTRACT** (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

In a military context where commanders have to move regularly out of their offices, and other static or mobile Command and Control facilities, the need to remain connected is of critical importance. Therefore, mobile technologies are vital for allowing commanders to maintain access and interact with command support tools and processes necessary for them to exercise their authority under a wide range of operating conditions.

A prototype called CHESS – Commander Handheld Support System, running on a wireless handheld device, was one of the important elements of a joint command decision support system for the 21st century (JCDS21). This prototype demonstrates how exchanges of information can occur between a wireless handheld device and Situational Awareness tools (COMMAND VIEW, Incident Management System), a planning system (Collaborative Operations Planning System) as well as decision support tools developed by JCDS21 TD. This report describes the main aspects of CHESS system design.

Dans un contexte militaire, où les commandants sont sujets à œuvrer à l'extérieur de leurs bureaux ainsi que de leurs postes de commandement statiques ou mobiles, le besoin de rester connecté est d'une importance critique. En conséquence, les technologies mobiles sont d'une importance capitale afin de permettre aux commandants de maintenir un accès et une interaction avec les outils d'aide au commandement et aux processus nécessaires pour exercer leur autorité pour un large éventail de conditions opérationnelles.

Un prototype, appelé CHESS – Commander Handheld Support System, fonctionnant sur un appareil de poche sans fil, était un des éléments important du système d'aide à la décision pour le vingt-et-unième siècle (JCDS21). Ce prototype démontre comment les échanges d'information peuvent être faits entre un appareil de poche sans fil et les outils d'éveil situationnel (COMMAND VIEW, Incident Management System), un système de planification (Collaborative Operations Planning System) ainsi que les outils d'aide à la décision développés dans le projet de démonstration technologique JCDS21. Ce rapport décrit les aspects principaux du design du système CHESS.

14. **KEYWORDS, DESCRIPTORS or IDENTIFIERS** (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

command and control; situation awareness; planning; decision support; handheld device; smart phone

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale



www.drdc-rddc.gc.ca