Overview of the Technology Candidates for the European Ground Systems Common Core (EGS-CC)

Mauro Pecchioli [1], Anthony Walsh [2],
European Space Agency, ESA/ESOC, Robert Bosch Strasse 5, 64293 Darmstadt, Germany

Juan María Carranza [3]
European Space Agency, ESA/ESTEC, Keplerlaan 1, 2201 AZ Noordwijk ZH, The Netherlands

Marie-Claire Charmeau [4]
CNES, Centre spatial de Toulouse, 18 avenue Edouard Belin, 31401 Toulouse Cedex 4, France

Michael Geyer [5]
DLR, Münchner Straße 20, 82234 Weßling, Germany

Pascal Parmentier [6]
EADS Astrium Satellites, 31 rue des Cosmonautes, Z.I. du Palays, 31402 Toulouse Cedex 4, France

Johannes Rueting [7]
EADS Astrium Space Transportation, P.O.Box 28 61 56, 28361 Bremen, Germany

Wolfgang Bothmer [8]
OHB System, Universitätsallee 27-29, 28359 Bremen, Germany

Pierre-Yves Schmerber [9]
Thales Alenia Space France, 100 boulevard du Midi, BP 99 - 06156 Cannes, France

Paolo Chiroli [10]
Thales Alenia Space Italy, S.S. Padana Superiore, 290, 20090 Vimodrone (MI) - Italia

The European Ground Systems – Common Core (EGS-CC) is a European initiative to develop a common infrastructure to support space systems monitoring and control in pre- and post-launch phases for all mission types. This will bring a number of benefits, such as the seamless transition from spacecraft Assembly, Integration and Testing (AIT) to mission operations, reduce cost and risk, support the modernisation of legacy systems and promote the exchange of ancillary implementations across organizations. The initiative is being undertaken as a collaboration of ESA, European National Agencies and European Prime Industry. In this paper we describe the main objectives of the EGS-CC initiative, the overall system concept and the features it will provide.

[1] EGS-CC system engineering team chair, mauro.pecchioli@esa.int
[2] EGS-CC system engineering team member, anthony.walsh@esa.int
[3] EGS-CC system engineering team deputy chair, juan.carranza@esa.int
[4] EGS-CC system engineering team member, marie-claire.charmeau@cnes.fr
[5] EGS-CC system engineering team member, michael.geyer@dlr.de
[6] EGS-CC system engineering team member, pascal.parmentier@frsat.astrium.eads.net
[7] EGS-CC system engineering team member, johannes.rueting@astrium.eads.net
[8] EGS-CC system engineering team member, wbothmer@ohb-system.de
[9] EGS-CC system engineering team member, pierre-yves.schmerber@thalesaleniaspace.com
[10] EGS-CC system engineering team member, paolo.chiroli@thalesaleniaspace.com

I. Introduction

A Background

Within Europe there are many different systems for monitoring and control used by companies/agencies for space system operations and Assembly Integration and Testing (AIT). Some of these systems are common to both operations and AIT, while some are specific. Often multiple systems are used in AIT of a space system by different companies or at different levels (e.g. payload/system) or in different phases. Many of these existing systems have reached or are reaching their end of life. The systems are often using old software technologies and hardware platforms that are difficult to modernise. The maintenance and evolution costs are therefore becoming excessively complex with time. The compatibility/exchange of information with other systems is also often difficult leading to little synergy across missions and project phases

Given the difficulties mentioned above, during 2009-2010, the European Space Agency (ESA) discussed with large European System Integrators, including Astrium Satellites, Astrium Space Transportation, Thales Alenia Space (France and Italy) and OHB System, the possibility of a collaboration to develop a European Ground Systems Common Core (EGS-CC) which would provide a common infrastructure to support space systems monitoring and control in pre- and post-launch phases. The French and German national space agencies, CNES and DLR also signalled their desire to join the initiative and a Memorandum of Understanding was finalised in support of the EGS-CC initiative.

B Technology Selection Process

The agreed development approach for the EGS-CC is that suitable technologies are reused where possible. This will reduce the effort required to implement the EGS-CC while also taking advantage of modern software approaches and high quality products. This however must also consider the dangers of technology lock-in and the unavoidable obsolescence of hardware and software products. A key decision for the EGS-CC is therefore the selection of technologies that will be used in the implementation. In this paper we present the approach taken for the selection of technologies and the current status of this process.

The process for the selection of technologies for EGS-CC is being performed according to the following steps.

1. Identify the technology domains which are considered applicable to the EGS-CC
2. Identify the criteria against which candidate technologies within a technology domain shall be assessed
3. Perform an assessment of candidate technologies and associated products, including the overall best integrated selection of candidates
4. Based on the previous points, finalise the technologies and products to be used for the implementation phases of the EGS-CC.

Two initial decisions were taken to scope the technology selection:

- **Operating Systems**: Linux is the selected Operating System for all EGS-CC implementations. In addition, compatibility with Windows is required for the execution environment front-end (user interface), for the preparation environment and for the development environment. Compatibility with Windows is also desirable for the execution environment back-end. Operating System independence should be achieved where possible.
- **Programming languages**: Java is the preferred language for all EGS-CC kernel components. This implies that all technology candidates have to offer a native interface towards Java based implementations, but should not rule out the capability to develop and/or integrate components implemented in a different language, for whatever reason (e.g. performance hot-spots, integration of legacy implementations, etc.). Further, it does not represent a constraint on the implementation of the selected products themselves (e.g. a product can be implemented in C++, provided that it supports Java based implementations in its interfaces). Components which do not belong to the kernel shall be able to communicate with the kernel components although they may not be implemented in Java.

C Technology Domains

Two distinct environments have been considered to identify the required technology domains:

- The EGS-CC Run-Time Environment addresses the technologies that will be included in the EGS-CC kernel, reference implementation, or reference test environment, and that are needed to run the software.
- The EGS-CC Software Development Environment addresses the technologies that will be used for the development and maintenance of the EGS-CC, but that are not needed to run the EGS-CC software.

The technology domains are areas where 3rd party technologies are considered applicable for reuse within the EGS-CC execution and software development environment. They are a guide to the selection of technology candidates, but they should not prevent to select the best overall integration of the technologies. The list of domains is driven by the current EGS-CC system concepts and user requirements, and can be further extended if needed in the next EGS-CC design phases.

II. Technology Selection Criteria

A Criteria Model

The EGS-CC technologies' selection criteria model is based on an extended FURPS+ model. It comprises the basic FURPS+ criteria enriched with additional ones covering the specific EGS-CC concerns (i.e. costs, IPR/licensing…)

The FURPS model has been developed at Hewlett-Packard. The FURPS acronym stands for the five categories that cover the following aspects:

- Functionality - Feature set, Capabilities, Generality.
- Usability - Human factors, Aesthetics, Consistency, Documentation
- Reliability - Frequency/severity of failure, Recoverability, Predictability, Accuracy, Mean time to failure
- Performance - Speed, Efficiency, Resource consumption, Throughput, Response time
- Supportability - Testability, Extensibility, Adaptability, Maintainability, Compatibility, Configurability, Serviceability, Installability, Localizability, Portability

After several usages of the model, the + was added to the model acronym in order to emphasize various attributes:

- Design requirements
- Implementation requirements
- Interface requirements
- Physical requirements

The EGS-CC specific concerns lead to add the following criteria:

- Costs – Implementation costs, Possession costs, Configuration costs, Adaptations costs.
- Marketability - Licensing, Property rights, Export rights
- Security - Authentication, authorization, encryption

B Functionality

Functional requirements are also supported by non-functional requirements, which impose constraints on the design and implementation. This criterion addresses the capacity of the candidate technology to fulfil the level of quality of the functional requirements.

For example, this criterion shall assess the system capacity to perform its mission in due time, managing the routine dataflow and potential peaks.

C Usability

Usability qualifies to which extent the system can be easily operated and in particular "learnability" and the "user friendliness" from the end user point of view. Thus, it addresses human factors, aesthetics and consistency in the user interface and documentation. It may encompass the following aspects:

- Consistent and integrated Look & Feel,
- Support users with different levels of expertise,
- Customizable for different projects/missions,
- Efficient and robust,
- Easy to learn (including training aspects),
- Good level of the available documentation.

This criterion is a major concern for the EGS-CC as it has a direct impact on the operational costs (i.e. minimizing the number of operators).

D Reliability

This criterion assesses the he ability the candidate technology to support the EGS-CC in performing its required functions under stated conditions for a specified period of time.

This criterion encompasses the following aspects:

- Availability (the amount of system "up time"), amount of time between failures or the ratio of time when the system is operational and accessible when required for use
- Ability to recover from failure without further impact,
- Accuracy and exactitude of system calculations
- Integrity and consistency (no data loss neither corrupted)

However, the maintainability is also an important part of reliability engineering, as the latest may affect at least the two fist listed items.

A high level of availability is required for the EGS-CC system to meet the operational requirements for CGS and EGSEs. In particular, the spacecraft operations and tests use cases imply that the whole ground segment should have no single point of failure, especially during critical phases (VSVT, LEOP…).

E Performance

This criterion qualifies the capacity of the candidate technology to support the EGS-CC in fulfilling its required functions within a given set of constraints. For example this may comprise:

Response time for a given function (e.g. start-up and shutdown time),

- Duration of the completion time for a given function,
- Capacity; like throughput (rate of processing work), data bandwidth / transmission time, number of connected facilities,
- Optimization of computing resource(s),
- High availability of the system (e.g. recovery time).

In particular, a high level of performance is required for the EGS-CC system to meet the stringent requirements relative to parameter processing. This criterion shall assess the degree to which the system is able to perform its mission in due time, managing the routine dataflow and potential peaks.

F Supportability

This criterion addresses the various elements which contributes to easily supporting the EGS-CC during its lifetime

- Testability: High degree of test automation,
- Extensibility: New components expecting during long life time and to build end users' applications,
- Portability: Easiness with which the system can be transferred from one hardware or software environment to another. Needs to support mainstream OS and different OS version over product life time,
- Adaptability: Facilitate tailoring for different projects and system applications, open interfaces (interoperability with other systems). This also qualifies the capacity of the solution to adapt to potential internal or external changes, in a timely and cost-effective manner
- Maintainability: OSS approach, proven and main stream technologies. This criteria addresses the easiness with which the system can be modified/upgraded to correct faults, improve performance or other attributes, or adapt to a changed environment
- Compatibility: Follow ECSS and CCSDS standards, avoid niche or single-vendor solutions, use Commodity PC/workstation hardware, reduce OS dependencies,
- Configurability: Where needed but no more to keep things simple and efficient,
- Installability: Automated where possible, follow platform standards, remote deployment,
- Scalability: Very broad range from single laptop to control centre. This also addresses the ability of the system to improve its performance after adding hardware, proportionally to the capacity added.
- Community: Qualifies the development and users profiles of the candidate solution.

During the EGS-CC long lifetime, system evolutions will be necessary due to the unavoidable obsolescence of hardware and COTS. Therefore, key drivers to be considered for the EGS-CC design are its maintainability and portability. The need to adapt to various kinds of applications (end users' systems) and to be able to interface numerous legacy systems shall be considered. Therefore, the EGS-CC design shall be highly flexible, modular and scalable to accommodate different scenarios for EGS-CC physical layout and allow easy integration with existing systems and for future evolutions.

In addition, a maximized maintainability is fundamental for cost-effectiveness.

G The Model "++

- Design requirement: Specifies or constrains the options for designing a system. For example, a relational database may be explicitly required.
- Implementation requirement: Specifies or constrains the coding or construction of a system. Examples include required standards, implementation languages, operating systems and resource limits.
- Interface requirement: Specifies an external item with which a system must interact, or constraints on formats or other factors used within such an interaction.
- Physical requirement: Specifies a physical constraint imposed on the hardware used to house the system — shape, size, or weight, for example.

H Costs

This specific criterion is reflecting the qualitative cost assessment of the system implementation and deployment. It includes the following items:

- Implementation costs: Assess the impact of the candidate solution on the development costs (i.e. licenses, schedule and risks - Maturity).
- Possession costs: Includes both costs to own (license fee) and to use (maintenance fee) the selected solution,
- Configuration costs: Addresses the complexity to parameterize the selected technology for each EGS-CC based application (end user system),
- Adaptations costs: Concerns the potential cost impact on elements/component added to the EGS-CC in order to build the end user system.

I Marketability

This specific criterion includes the following items:

- Licensing policy: Licensing model and price policy applied to the candidate solution. The impact of the licensing scheme on the complete EGS-CC based application shall be assessed.
- IPR: This element qualifies the legal aspect attached to the use of the candidate solution and its impact on the EGS-CC business cases.
- Export rights: Identification of export restrictions and procedures. Assessment of the compliance towards the EGS-CC driving constraints (e.g. no export restriction)

The candidate technology shall ideally have at-least two viable open source implementations. Only open source license can be considered which do not contaminate the EGS-CC software itself with the same license conditions (i.e. viral). This does not exclude the use of licenses, such as GPL, but only when it does not contaminate the EGS-CC software which will have its own licensing conditions. Commercial products may be considered, but the selection must be fully justified (e.g. is the only product that could fully meet the requirements) and only if it is possible to easily replace the product with an alternative (perhaps with some loss of capability).

J Security

This criterion assesses the degree to which the system is able to protect its data and restrict its access to authorize users. It includes the following items:

- Authentication: Guarantees the origin of the incoming entities (e.g. configuration data, commands, service requests…),
- Authorization: This element is particularly addressing the capacity to filter user access to elementary functions. The selected technologies shall not put any restriction on this capability,
- Encryption: read protection of data (stored, exchanged).

This aspect shall be checked very soon as inappropriate solutions may jeopardize the capability to reach the expected level of security for the final EGS-CC based applications

III. EGS-CC Run-Time Environment

A Component Framework

The component framework supports the development, composition, deployment and execution of the EGS-CC software components, which form the building blocks of an EGS-CC system. The EGS-CC functional behaviour is implemented by components running within the component framework. The component framework provides the mechanisms by which the EGS-CC can be extended and customized, including a clean separation of communication protocols from component interfaces (via service bindings), dependency injection within the components, and the capacity to provide lightweight components (plain old java objects approach).

Identified Candidates are: OSGI, SCA (Service Component Architecture), EJB3 (Enterprise Java Beans 3), Spring, Corba Component Model (CCM), OPC Unified Architecture, Google Guice

The following technology candidates are selected for a detailed analysis against the selection criteria:

- OSGi
- Spring Framework

The list of candidates identified is considered complete and no further analysis of other candidates is considered necessary. OSGi has been identified by a number of SET and SME members as a very suitable technology for the component framework. The positive points of OSGi include:

- OSGi is a mature technology with a high level of adoption in middleware implementations.
- A number of high quality open source implementations are available, including those with liberal licenses (e.g. Felix, Equinox, etc).
- Good support for modularizations and plug ability, where variations and extensions of a core system can easily be accommodated.
- Remote services specification supports the externalization of services outside of the JVM.
- Blueprint Container specification provides a dependency injection framework for OSGi to support application level development (based on Spring Dynamic Modules).

Spring and EJB (JEE) are very mature with a large user community and dominate within the Enterprise Application domain. The experience of Spring and EJB within the SET and SME members is more limited than for OSGi, but both are considered suitable candidates which potentially can fulfil the EGS-CC requirements. EJB is however considered as being rather heavyweight within the JEE technology stack.

SCA is considered a good technology that can fulfil the EGS-CC component framework requirements. However the adoption level, availability of good open source implementations and the activity of the user community has been lower than initially expected and doubts remain over the long term relevance of SCA within the market.

The other identified component framework candidates are not considered suitable as their adoption levels compared to the candidates described above is low.

B Service Integration Platform

The service integration platform supports the integration of a complete EGS-CC based system and its integration with external systems using a service orientated approach.

The service integration platform provides the basis for the top level integration of a complete EGS-CC system. It includes various integration features, such as flow control, message transformation, etc. A critical feature is performance (data throughput and latencies) as there is an overhead going through the message bus itself and also the need for transformation logic to be performed.

With regards to integration of the service integration platform shall:

- be decoupled from the selected component framework to enable easier evolution of the two products,
- provide good integration of services of provides / consumers running within the component framework,
- be open to the integration of different communication protocols.

Identified Candidates are: Enterprise Service Bus (ESB), CCSDS MAL/COM, Apache Camel, Web Service Integration Frameworks, Eclipse Communication Framework (ECF)

The following technology candidates are selected for a detailed analysis against the selection criteria:

- The service integration platform shall be based on an Enterprise Service Bus (ESB)
- Support for mediation and transformation logic be based on Apache Camel

The list of candidates identified is considered complete and no further analysis of other candidates is considered necessary. ESB's have been clearly identified by the SET and SME's as a good solution for a platform in which integration logic can be deployed. No other candidates are therefore considered. Many ESB's open source products are identified and therefore a comparison against the selection criteria for the leading open source products is required. This analysis shall in particular take into consideration the key drivers:

- EGS-CC performance requirements
- Compatibility with the selected Component Framework
- Integration of CCSDS MAL/COM/MO Services
- Camel Integration

Camel is considered a powerful solution for mediation and transformation logic and therefore a combination of an ESB with Camel is recommended.

The CCSDS MAL/COM/MO Services are currently considered too immature and no known open source implementations available. They are therefore only considered in the context of external services. An ESB however can provide the platform in which CCSDS MAL/COM/MO services are implemented.

C Communication and Data Distribution

Support for the efficient distribution of messages and data within the EGS-CC, including the marshalling and serialisation of data.

Message-oriented middleware (MOM) is software or hardware infrastructure supporting sending and receiving messages between distributed systems. MOM allows application modules to be distributed over heterogeneous platforms and reduces the complexity of developing applications that span multiple operating systems and network protocols. The middleware creates a distributed communications layer that insulates the application developer from the details of the various operating system and network interfaces. APIs that extend across diverse platforms and networks are typically provided by MOM

In addition, many inter-application communications have an intrinsically synchronous aspect, with the sender specifically wanting to wait for a reply to a message before continuing (see real-time computing and near-real-time for extreme cases). Because message-based communication inherently functions asynchronously, it may not fit well in such situations. That said, most MOM systems have facilities to group a request and a response as a single pseudo-synchronous transaction.

Identified Candidates are: Java Messaging Service (JMS), Advanced Message Queuing Protocol (AMQP), OMG Notification Service, OMG Data Distribution Service (DDS), Google Protocol Buffers, Tibco Rendezvous, Apache Thrift, MessagePack, ZeroMQ, ASN.1

The list of candidates is considered complete and no further analysis of other candidates is considered necessary. The following technology candidates are selected for a detailed analysis against the selection criteria:

- Java Messaging Service (JMS). The analysis of JMS shall be performed on the leading open source JMS products.
- ZeroMQ.

If neither of the above candidates are evaluated to be suitable, an analysis of OMG Data Distribution Service (DDS) analysis shall also be performed.

No clear favoured candidates for the encoding technology (i.e. Google Protocol Buffers, MessagePack and ASN.1) to efficiently encode/decode messages passed through the messaging technology have been identified.

JMS has been identified by many SET and SME members as a suitable messaging technology. Some of the advantages of JMS include:

- very large user base
- mature open source implementations (e.g. Apache ActiveMQ, Apache Apollo, Joran, HornetQ, etc)

- Bindings for other languages supported by implementations, although not standardized

An analysis of the different open source JMS implementation is required, especially with regard to performance. Doubts exist by some SET members with regard to fulfilling all EGS-CC performance requirements. For this reason, ZeroMQ which is a very light weight messaging product shall also be considered within the analysis.

AMQP has also been identified and while it has some adoption, is less mature than JMS with fewer open source implementations.

The OMG Data Distribution Service is an interesting technology for the real-time distribution of data (e.g. telemetry data). It is however a rather niche technology and the available open source implementations are immature. The additional benefit of using DDS compared with other approaches for the specific EGS-CC requirements is not clear and therefore only considered for analysis if the other candidates are evaluated unable to fulfil the EGS-CC selection criteria.

D System Run-time Management

Support a common approach for the management and administration of the EGS-CC run-time system and legacy platforms. The purpose is to provide a harmonised approach to run-time management of all the systems that comprise the EGS-CC.

Identified Candidates are: Java Management Extensions (JMX), ZooKeeper

Java Management Extensions (JMX) is selected for the detailed analysis for the EGS-CC system run-time management. No other suitable technology candidates for system run-time management have been identified. JMX must be supported by the applicable technology domain if it is to be efficiently utilised. The technology candidates for the at-least the following domains shall therefore also be analysed for their support for JMX:

- Component Framework
- Service Integration Platform
- Communication and Data Distribution

Java Management Extensions is the only standard approach within the Java context. The list of candidates is therefore considered complete, as no other viable candidates are known to exist. Products that are to be managed using JMX must be JMX aware and provide the necessary JMX support.

E Logging and Tracing

Logging and tracing of events and messages within the EGS-CC.

In the System Context document in the context of messaging. Although messaging and logging are connected, we address them separately with regards to technology assessment and describe the relationship to both messaging and data archiving in general.

A number of logging frameworks exist and are candidates for selection for the logging solution within the EGS-CC. Many of these frameworks breakdown logging into three logical components (see Wikipedia):

- Logger – this is responsible for capturing the message to be logged, including any associated metadata and passing it to the logging framework. The logger provides the interface with EGS-CC components which are the source of log messages.
- Formatter – the logging framework uses a format to take the log message (including meta-data) and formats its output. This enables a clean separation between the context of a log message and how the message is presented in its output form. This allows the format to be easily changed by either replacing the formatter or by changing the configuration settings of an existing formatter. Different formatters could also be used depending of the message type.
- Handler – the logging framework passes the formatted message a handler for disposition. The handler could be responsible for sending the formatted to a display, writing it to a file / database for storage, etc. The selected handler will depend on the message (e.g. message type) and more than one handler can be invoked for a given message. For example, it could be that the log framework is configured that trace messages are only processed (for performance reasons) by the local log manager.

Identified Candidates are: Apache Log4j, Simple Logging Facade for Java (SLF4J), OSGi Logging Service, OMG Logging Service

Apache Log4j and the Logback successor project are selected for analysis against selection criteria and if non compliances are discovered, then the analysis can ve widen to consider other logging frameworks. The benefit of using SLF4J as the logging access layer with the EGS-CC application layer shall also be investigated,

Apache Log4j has been identified as the prime candidate by SME and SET members given its high adoption levels. The ability to fulfil all EGS-CC logging requirements still needs to be verified and if the detail analysis identifies issues, then other logging frameworks shall be considered.

F Data Persistence

Covers the framework used for the persistence of data within the preparation and run-time environment. This covers M&C data definitions, software configuration data, software component state, etc. It allows to provide an harmonized approach to data access, while supporting different data access back-ends.

Data persistence attributes the reliability of parts of the EGS-CC system. An increased reliability follows from the fact that state can be restored even after a restart of a component/program/subsystem/system.

Which objects are to be managed persistently needs to be determined. It is already safe to state that configuration data objects for example are good candidates for this.

Identified Candidates are: Service Data Objects (SDO), Java Data Objects (JDO), Java Persistence API (JPA), Hibernate, JDBC, RDBMS, NoSQL, Eclipse EMF, Ehcache

It is recommended that the data persistence domain is split into two sub-domains; data access and data storage. The list of candidates for the data access sub-domain is considered complete and the following is selected for detailed analysis against the selection criteria:

- Java Persistance API (JPA)
- Java Data Objects (JDO)
- Service Data Objects (SDO)
- Eclipse Modelling Framework (EMF)

It is recommended that the selection of data storage technologies is delayed until the EGS-CC design phase.

The SME and SET members have no clear consensus on the selection of the data persistence technology. It is recognised that the data persistence should be split between the data access layer and the data storage layer. This enables to change the back end storage technology while maintaining the same interface (i.e. data access layer) with the EGS-CC application layer. The data access layer is therefore strongly coupled into the design of the EGS-CC application layer and needs to be selected now.

Hibernate is considered too specifically oriented toward access to relational database systems.

G Data Archiving

The efficient storage and retrieval of time ordered data generated during an EGS-CC system run-time session. Performance is a key driver and where the complete history of all samples is stored.

Identified Candidates are: RDBS Based Solutions, HDF5, Key Value Datastores (NoQSL)

The following technology candidates are selected as the basis for the data archiving implementation:

- RDMS
- NoSQL (recommended products Tokyo Cabinets / Kyoto Cabinets and LevelDB)
- HDF5

As the internal implementation of the data archive is transparent to the other EGS-CC components, the key issue for the selection of technology shall be performance, especially for the storage and retrieval of the MCM state. As an EGS-CC project risk mitigation exercise, the fulfilment of the data archive performance and storage requirements should be demonstrated by a proof of concept with the selected data archiving technology.

There is no clear consensus on which technology should be used for the internal implementation of the data archive. Both NoSQL and RDBS technologies have both been reported by SET members to have been used in the successful implementation of data archives.

H Data Modelling and Editing

Support for the development of data models and the editors of data compliant with these data models.

A data modelling framework shall be used to support the model driven approach described in the EGS-CC Conceptual Data Model. It is expected to support the efficient development of tools used in the EGS-CC Reference Test Environment to edit / utilize / store data compliant with the EGS-CC Data Model

Identified Candidates are: Eclipse Modelling Framework (EMF), UML Modelling, Fact Based Modelling

The Eclipse Modelling Framework (EMF) is selected as the best candidate for data tooling and no other suitable candidates are considered for further analysis against the selection criteria. Reasons for selecting EMF include:

- Mature and well adopted within industry
- Has already been successfully used by SME and SET members
- Is compatible with the proposed Rich Client UI framework (i.e. Eclipse RCP)

The EGS-CC data model in phase A of the EGS-CC project is captured using UML. For phase B onwards it is recommended that the suitability of the following data modelling approaches is considered with the constraint that the conceptual data model can be easily transformed and utilised by EMF for the data tooling:

- UML
- Object Role Modelling (ORM)
- eCore

I File Management

Management of files, including features not supported by standard file systems (e.g. versioning, file sets management, files distribution and remote access) and file content (e.g. compliance with applicable schemas).

It is probably not possible to select a technology that fulfils all the requirements, but should focus on those functions for which general solutions are known to exist (i.e. Revision Control System -> SVN, GIT, etc)

Identified Candidates are: Revision Control Systems (SVN, GIT, Mercurial), Java Content Repository (JCR).

The list of candidates is considered complete and no further analysis of other candidates is considered necessary. The following technology candidates are selected for a detailed analysis against the selection criteria :

- Apache Subversion (SVN)
- GIT
- Content Repository API for Java (JCR)

The SME and SET members have identified SVN and GIT as popular revision control systems. SVN uses a client-server model while GIT uses a distributed model where each client maintains there own copy of the repository which is periodically synchronised with the other distributed repositories. It is unclear which model is best suited for EGS-CC. It is also unclear if SVN or GIT could fulfil all the file management requirements without additional functionality needing to be being implemented.

The Java Content Repository has been identified as a potential technology for file management. The JCR provides a more generalised concept for the storage and management of data than a revision control system, but it is not clear if the extra complexity is worth the added value. The maturity of open source JCR content repositories such as Apache Jackrabbit needs further analysis.

Given the initial analysis of SVN, GIT and JCR, it is recommended that all three are analysed in detail against the selection criteria. As part of this analysis the following shall also be considered:

- The ability to combine SVN and GIT so that SVN can be used as a central repository and GIT for local repositories.
- Ensure that the GIT does not also infect clients (i.e. EGS-CC applications) with GPL license conditions.

J Scripting Language

The general scripting language used within the EGS-CC run-time system.

Identified Candidates are: Javascript, Python, Groovy, PERL.

It is recommended that EGS-CC should support scripting languages that are compliant with the JSR-223 specification. This would allow users to select the most appropriate language for their EGS-CC system deployment. This recommendation should be verified against the selection criteria for two or more script languages and demonstrated by a proof of concept.

A number of scripting languages have been identified by SET members and SME's, but no clear consensus is identified. There are many scripting languages available and the choice of language is also a question of preference of the user community for which no information currently exists.

Given the lack of clear consensus, the selection of script engine by users for a specific EGS-CC system deployment would be the recommended approach. The JSR 223 "Java Scripting API" allows the integration of different scripting languages within an application. The application is therefore able to host and interact with any JSR 233 compliant script engine. The following JSR-223 compliant script engine implementations of script languages identified by SME's and SET members are known to exist:

- JavaScript : Apache Rhino 1.6R7 (http://www.mozilla.org/rhino/)
- Groovy: Groovy 1.5.4 (http://groovy.codehaus.org/)
- Python : Jython 2.2.1 (http://jython.sourceforge.net/Project/index.html)

The ability for the EGS-CC to host and interact with any JSR 223 compliant script engine should be demonstrated by a proof of concept.

K Procedure Language

Language used by users of the EGS-CC for operations and test execution automation. The automation language shall be compliant with the ECSS E-70-31 requirements.

Identified Candidates are: ECSS-E-ST-70-32C compliant language (such as PLUTO), MOIS, Java

There is no clear consensus for the procedure language and the following identified candidates are selected for further detailed analysis against the selection criteria.

- ECSS-E-ST-70-32C compliant language (specific for spacecraft operations and tests)
- Java (dynamic compilation)
- Reuse of script language (see previous section)

For each of the above, the analysis should also determine if:

- It provides a suitable exchange format or is compatible with commonly accepted exchange formats
- A suitable execution engine implementation exists
- Addition views of the language syntax (i.e. graphical, tabular) can easily be accommodated in the preparation and execution environments.

No clear consensus exists on the choice of procedure language and therefore a detailed analysis of all identified candidates should be performed with the exception of MOIS which is a commercial product (although MOIS can be considered within the context of a procedure preparation environment which is outside of the scope of the EGS-CC).

L Expression Language

Language used for the evaluation of expressions (e.g. synthetic parameter evaluation).

Require analysis to identify candidate list, after which a selection can be performed. It should be analysed if the selected scripting language / procedure language (see previous sections) can fulfil the expression language requirements. However, the capability of this language to deal with proper operations with parameter values (taking into account their type, validity, state, etc) shall be carefully assessed.

There have been no inputs from SME and SET members concerning expression language candidates other than a desire to reuse if possible the selected script / procedure language expression capabilities.

M User Interface

User interface framework used for EGS-CC clients. Depending on the type of client (e.g. smart clients, thin clients, remote clients) different technological solutions may be employed.

Identified Candidates for rich clients are: Eclipse Rich Client Platform (RCP), Swing, Qt, Netbeans.
Identified Candidates for thin clients are: AJAX, FLEX, Jetty, VAADIN Java Framework, LifeRay Portal, Google Web Toolkit (GWT).

The user interface framework used for rich and thin clients can be different, however the possibility to maintain the same look and feel is important.
The rich client interface framework is considered complete and only Eclipse RCP is seleted for further detailed analysis against the selection criteria.

Eclipse RCP has been identified and successfully adopted by a large number of SME and SET members. Although other Java based GUI technologies were identified, the strong preference for Eclipse RCP is clear and it is only considered necessary to verify that Eclipse RCP can fulfil the selection criteria. Qt is also C++ based which goes against the criteria of using Java as the baseline language.

A number of technology candidates are identified for the thin client framework, but it is recognised that the technologies within this domain are evolving very rapidly. As it is not considered necessary to select a thin client framework at this time, it is recommended to delay this decision until the EGS-CC implementation phase when a selection based on the most up to date state of the thin client technologies can be made.

N User Defined Displays

Technology used to create and render displays according to user definitions (e.g. graphical plots, synoptic displays, etc).

Identified Candidates are: Scalable Vector Graphics (SVG), X3D, Charting Libraries (JfreeChart).

There was little input by SME and SET members for User Defined Displays and therefore further analysis of this technology domain is considered necessary to complete the candidate list.

It is recommended to split the User Defined Displays domain into display data formats (e.g. SVG) and graphics libraries (e.g. JFreeChart). These should be compatible with chosen UI framework (i.e. Eclipse RCP) and cover:

- Charts and Graphs
- Synoptic (Mimic) Displays
- 3D Displays

It is considered that the selection of user defined displays technologies can be deferred until the design phase.

O Security

Security handling, including the authentication and authorisation of users and its integration within the EGS-CC system.

Identified Candidates are: JAAS, Lightweight Directory Access Protocol (LDAP), Apache Shiro, OpenEM, Progress Actional, Codelogin, Whitelisting.

The selection of security technologies is deferred until the design phase, as it is linked to the framework and other technology selections.

P Post Processing & Reporting

Off-line analysis of data generated by the EGS-CC system and associated reports generation (according to user layout definitions).

Identified Candidates are: Eclipse Business Intelligence and Reporting Tools (BIRT), JasperReport, .

For plotting it is considered very desirable to have the same solution for runtime and post processing environments. This implies that basic data analysis and reporting capabilities (e.g. plotting of statistical data, zooming in/out, and combined displays of parameter and event occurrences data) shall be supported by the plotting displays.

There was limited input by the SME and SET members for post processing and reporting tool candidates. The prime candidate identified is Eclipse Business Intelligence and Reporting Tools (BIRT) which is well adopted within industry with a large user based. BIRT also provides compatibility with Eclipse RCP.

BIRT is therefore selected for analysis against both the detailed selection criteria and the ability to use the same plotting solution as adopted for the run-time display environment.

IV. EGS-CC Software Development Environment

The EGS-CC Software Development Environment shall

- Support distributed development
- Provide state-of-the-art development environment
- Support open and large community
- Minimise operations and per-developer costs
- Operate for 20 years
- Be small and simple

The following two figures visualize a possible distributed environment with a central repository and a central build server (first figure) and an example workflow on this environment (second figure).

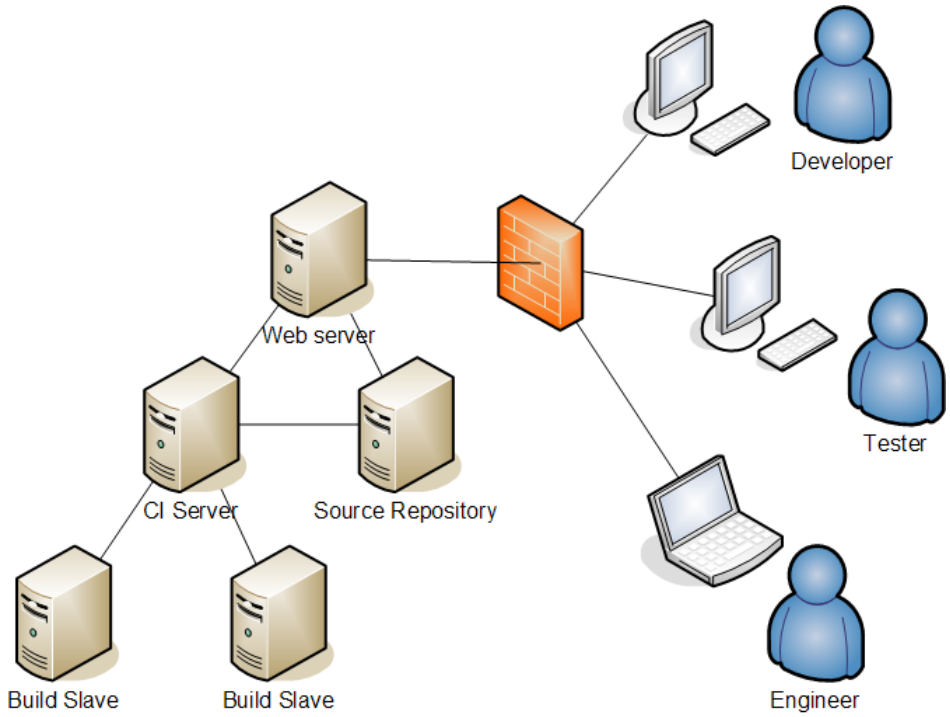

Figure 1. Distributed Development Environment

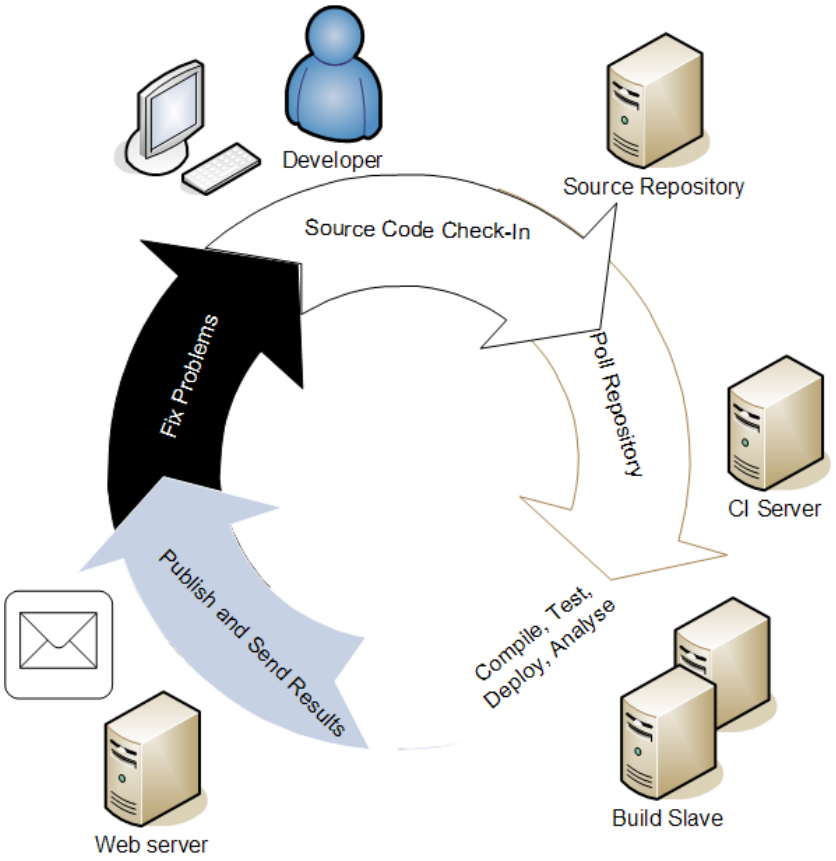

Figure 2. Development Workflow

The selected technology candidates integration capacity has been demonstrated in the frame of the AITS project distributed software development environment

A Requirements Management

Support for the management of EGS-CC requirements throughout the project life-cycle.
The selected technology for use starting in phase A is DOORS (commercial)

B Integrated Development Environment

Provides comprehensive facilities for software development, such as source code editor, compiler, debugger and which supports the integration of other SDE facilities into a single environment.
The selected technology for use starting in phase B or C is Eclipse

C Design Language and Tooling

EGS-CC design language and the supporting tools used for the EGS-CC design and the generation of documentation and software artefacts.
The selected technology for use starting in phase A is UML Enterprise Architect (commercial)

D Configuration Management

Supports the storage, configuration and change control of the EGS-CC software and documentation.
The selected technology for use starting in phase A is Subversion

E Build System

The building of deployable EGS-CC system artefacts from EGS-CC sources (e.g. generation of binaries from source code).

The selected technology for use starting in phase B or C is Maven with Nexus and Java Development Kit (JDK)

F Installation and Deployment Management

Support for the installation and deployment of the EGS-CC into the target run-time environment.

The selected technology for use starting in phase B or C is Maven with RPM

G Testing Framework

Supporting test framework(s) used for the verification and validation of the EGS-CC system at various levels (i.e. unit, integration and system) according to ECSS E-40.

The selected technology for use starting in phase B or C is JUnit with TestNG (commercial)

H Software Quality Measurements

Support the production of EGS-CC software quality metrics (e.g. compliance against coding standards, requirements coverage, code coverage, performance, etc).

The selected technology for use starting in phase B or C depend on the selected quality standards, and as such the initial selection can be completed with additional tools later: Sonar, Checkstyle, Findbugs

I Continuous Integration

Implement the continuous processes of applying quality control within the EGS-CC development (i.e. small pieces of effort, applied frequently).

The selected technology for use starting in phase B or C is Jenkins

J Issue Tracking and Reporting

The tracking and reporting of issues during the EGS-CC development and maintenance.

The selected technology for use starting in phase A is Jira (commercial)

K Code Reviews

Support for code reviews within a distributed environment.

The selected technology for use starting in phase B or C is Crucible (commercial)

L Document Authoring

Support for technical documentation production (e.g. User Manuals, Training Material, etc) so that document content can be created in a presentation-neutral form and published in a variety of formats, such as HTML or PDF.

The selected technology for use starting in phase B or C is Confluence (commercial) with Microsoft Office (commercial) and possibly DocBook

M Project Web Portal

Central web-portal portal where all information related to the EGS-CC can be managed and shared, including support for mailing list and a Wiki to promote communication and the sharing of knowledge within the EGS-CC development team.

The selected technology for use starting in phase A is Confluence (commercial)

N Repository Browser

Support for repository management within a distributed environment, where source code and documentation can be accessed.

The selected technology for use starting in phase B or C is FishEye (commercial)

O Test Management & Acceptance

The selected technology for use starting in phase B or C is SpiraTest with Jubula.

P Source Code Documentation

Support for adding and extracting code documentation within source code files.

The selected technology for use starting in phase B or C is Javadoc.

Q Mailing Lists

The selected technology for use starting in phase A is Mailman.

R User Management

The selected technology for use starting in phase A is Atlassian Crowd (commercial)

V. Conclusions

An initial selection of potential candidate technologies and products has been performed, taking into account inputs provided by Small and Medium Enterprises (SMEs) and Prime industry. After an initial assessment of the most promising candidates, a detailed assessment of these candidates will soon be undertaken from which an overall best integrated selection of candidates will be chosen and used as input to the implementation phases of EGS-CC. This detailed assessment will be performed within the context of a study to be performed within ESA's Basic Technology Research Programme (TRP).