Missouri University of Science and Technology
Scholars' Mine

Mechanical and Aerospace Engineering Faculty Research & Creative Works

Mechanical and Aerospace Engineering

01 Jan 2006

Analysis of Damage in Laminated Automotive Glazing Subjected to Simulated Head Impact

Shuangmei Zhao

Li Chai

S. D. Barbat

Lokeswarappa R. Dharani
Missouri University of Science and Technology, dharani@mst.edu

Follow this and additional works at: https://scholarsmine.mst.edu/mec_aereng_facwork

 Part of the Aerospace Engineering Commons, and the Mechanical Engineering Commons

Recommended Citation

S. Zhao et al., "Analysis of Damage in Laminated Automotive Glazing Subjected to Simulated Head Impact," *Engineering Failure Analysis*, Elsevier, Jan 2006.
The definitive version is available at https://doi.org/10.1016/j.engfailanal.2004.12.038

This Article - Journal is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Mechanical and Aerospace Engineering Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

Paper #162

Modeling Net-Centric System of Systems using the Systems Modeling Language

Madwaraj Rao, Sreeram Ramakrishnan and Cihan Dagli
Madwaraj Rao, Sreeram Ramakrishnan and Cihan Dagli*
Smart Engineering Systems Laboratory, Engineering and Systems Engineering Dept.,
University of Missouri – Rolla, MO 65409-0370
murwz9@umr.edu, sreeram@umr.edu, dagli@umr.edu

Abstract

Understanding the operations of a large 'net-centric system-of-systems requires in-depth knowledge of the interfaces among the various systems, sub-systems and components. Architectural modeling can help in reducing the complexity involved in designing such large networked systems. An example of such a complex system is the Global Earth Observation System of Systems (GEOSS) – a system for monitoring and collecting information related to Earth's resources. This paper demonstrates the use of Systems Modeling Language (SysML), which supports specification, analysis, design, verification and validation of a broad range of complex systems, to model some aspects of the GEOSS. The paper discusses issues related to architecture description, development, presentation, and integration for the chosen domain. The paper discusses issues related to model evaluation and how architectures can be simulated to better understand their efficacy.

Introduction

Earth quakes and Tsunamis may be predictable with better accuracy with the help of an observation system to provide early warnings about their occurrences. GEOSS is an evolving system, whose purpose is to understand the earth, including its weather, climate, oceans, land, eco systems, geology, natural resources and natural and human induced hazards. GEOSS will consist of remotely sensed and in situ systems. From a functional view, GEOSS includes the following components: a component to acquire observations based on existing national, regional and global systems called the Observing component; a component to process data into useful information called the Data Processing and Transforming node; and a component required to exchange and disseminate observational data and information called the Data Producing Component. The data exchange and dissemination component will provide the necessary data and access to designated centers, including those for archive and for on-demand access. It will help in data and information sharing by supporting interoperability. The communication among the various systems in the GEOSS will happen with the use of technologies such as the Internet. Data could be exchanged among satellites on orbit or disks could be sent by mail from remote locations. Information could be broadcast using television or could be displayed on highways. Further information about the GEOSS architecture can be obtained from [Group on Earth Observations, 2005].

Modeling

This paper follows a Model Driven Architecture development process. Model Driven Architecture (MDA) development has its own merits and is discussed by [Hause et al. 2004] in detail with the usage of various UML diagrams. [Hause et al. 2004b] discuss the MDA initiative from OMG to overcome the shortcomings of UML for systems engineers. Systems engineers have been using UML to model systems but have been handicapped by the limitations of UML for modeling non-software systems. [Hause et al 2001] discuss the required extensions to UML in order to model non-software systems. SysML was formulated to overcome the shortcomings of UML. In this paper, the GEOSS has been modeled based on SysML 1.0A. SysML diagrams can be grouped into structural models and behavioral models. The use case diagrams, activity diagrams and sequence diagrams are utilized to build the behavioral model of a system. The block definition diagram is useful to build the structural model of the system. The behavioral model reflects how the system operates by modeling the interactions within a system. The structural model shows the entities of the system and the relationships between them.

The modeling methodology employed in [Lee et al. 2003] uses UML to model an information system whereas we have used SysML to model a System of Systems (SOS). Apart from modeling the GEOSS, the advantages that SysML can offer to model systems as compared to UML 2.0 is highlighted in this paper. The first step in the modeling is capturing requirements. The use cases for the system can then be developed from the requirements. The use case diagram is the first step in developing a scenario between the users and the system. Once the scenarios for the use cases are developed, they can be further conveyed using the activity diagram. The activities in the activity diagram are then time sequenced using the sequence diagram. Finally, the block definition diagram provides a static picture of the system and its interactions.

Requirements Diagrams

The requirements phase is, typically the first phase of the systems engineering life cycle model. The design and development of any system begins with the understanding of the end user requirements. The diagram that is most frequently used for modeling requirements in the Unified Modeling Language (UML) is the use case diagram. A number of methods have been used previously by systems engineers to model requirements, a typical example detailed in [Daniels et al. 2004] describes a hybrid process of combining traditional requirements, which are usually in the form of shall statements, and use cases. [Holt 2004] describes the requirements management process in detail. The high level requirements are listed down and they are grouped under business, functional and non-functional requirements. Business requirements address business concerns such as schedule and cost. Functional requirements are the user requirements for a system defining the desired functionality of the system. Non-functional requirements constrain the functional requirements. However, in SysML, requirement diagrams are used to model requirements. Requirements can be deduced from other requirements using the <<derive>> relationship. A requirement can be fulfilled by other model elements using the <<satisfy>> relationship. A requirement can be verified by various behaviors using the <<verify>> relationship. All these have been developed from the UML <<trace>> relationship. Each requirement is described using a text ‘shall’ statement. Requirements can be usually listed after interviewing the stakeholders. The requirements need to be properly documented and the ‘text’ statements have to be revisited iteratively until they are confirmed to convey the requirement

clearly. The requirements are usually managed using a tool such as DOORS. In DOORS the 'shall' statements can be listed as objects in a module and links can be provided to the other related requirements which have one of the relationship such as <<derive>>, <<satisfy>> and <<verify>> with the other requirements or design parameters. Such an exercise can help in building the requirements model smoothly thus helping in conveying the message to the stakeholder or end user clearly.

The requirements model for the GEOSS is as shown in Figure 1 and Figure 2. The high level requirements for the GEOSS were drawn from [Group on Earth Observations, 2005]. A <<derive>> relationship has been used to model this dependency. The high level requirement of GEOSS, which is to launch successful Earth observations, can be decomposed into sub requirements based on the type of observation needed. These requirements are all functional requirements. One of the high level requirements of the GEOSS is to 'Reduce loss of life and property from natural and human-induced disasters' which is given an ID: 101. The other attributes of the requirement such as source, text, kind, verifyMethod and risk are filled in as shown in Figure 1. The system containing the Registry of Environmental Protection Agency's Applications and Databases (READ), with ID:201 in Figure 1, fulfills one part of the requirement. A <<satisfy>> relationship has been used to model this dependency. The requirements shown in Figure 1 are listed below with their IDs [Group on Earth Observations, 2005]:

101: Reduce loss of life and property from natural and human-induced disasters
102: Understanding environmental factors affecting human health and well being
103: Improving management of energy resources

The descriptions of systems which satisfy these requirements are listed in [Refer: http://www.epa.gov/geoss], One such example is given below:

201: The READ is an authoritative registry that uniquely identifies the Environmental Protection Agency's (EPA) diverse information resources, including computer application systems, databases and models

One of the non-functional requirements – 'The data rate for exchange of electronic data between any two nodes of the GEOSS shall be 100 Mbps (Mega bits per second)' is considered to be a high level requirement and is modeled as shown in Figure 2. It is easier to model the non-functional requirements if a <<constrain>> stereotype can be defined in SysML. The non-functional requirement can then be modeled as a constraint to the functional requirement.

A scenario may be defined as a sequence of interactions between a user and a system. For the GEOSS, a scenario was selected from [Group on Earth Observations, 2005]. The scenario was revised as required for modeling purposes and is as worded in Figure 3. The scenario is related to the high level requirement of the GEOSS, which is to "Reduce loss of life and property from natural and human induced disasters". The scenario conveys how GEOSS could help nations save lives in the event of an earthquake. Scenarios are descriptions of the use cases for the system and the use cases for the scenario in Figure 3 are as shown in Figure 4 and 5. Figure 5 further elaborates the 'Collect Observed Information' use case. The top level "Gather Tsunami Data/Information" base use case is decomposed using <<include>> relationship to define new use cases that are performed as part of the base use case.

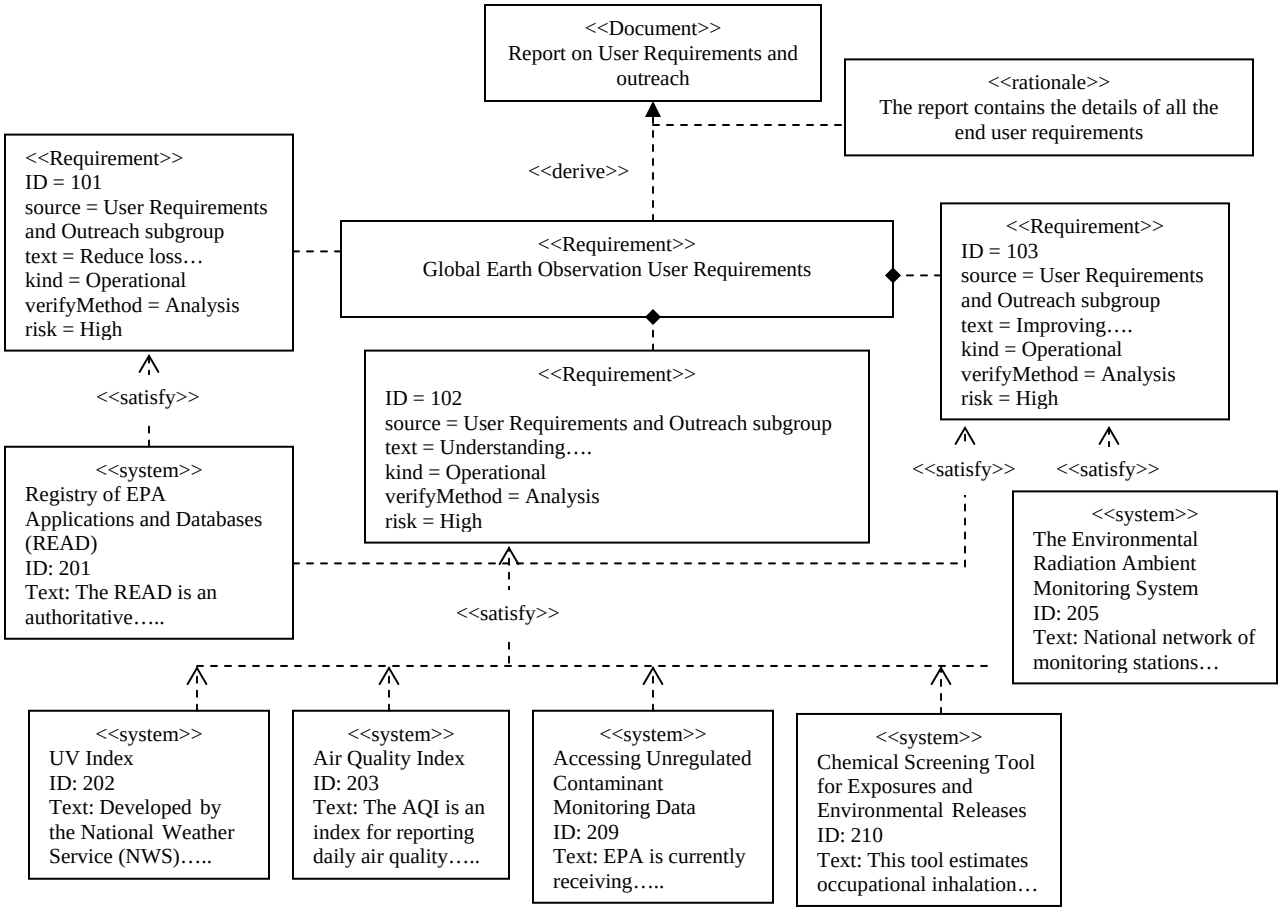

Figure 1. Requirements Diagram (I) of the GEOSS

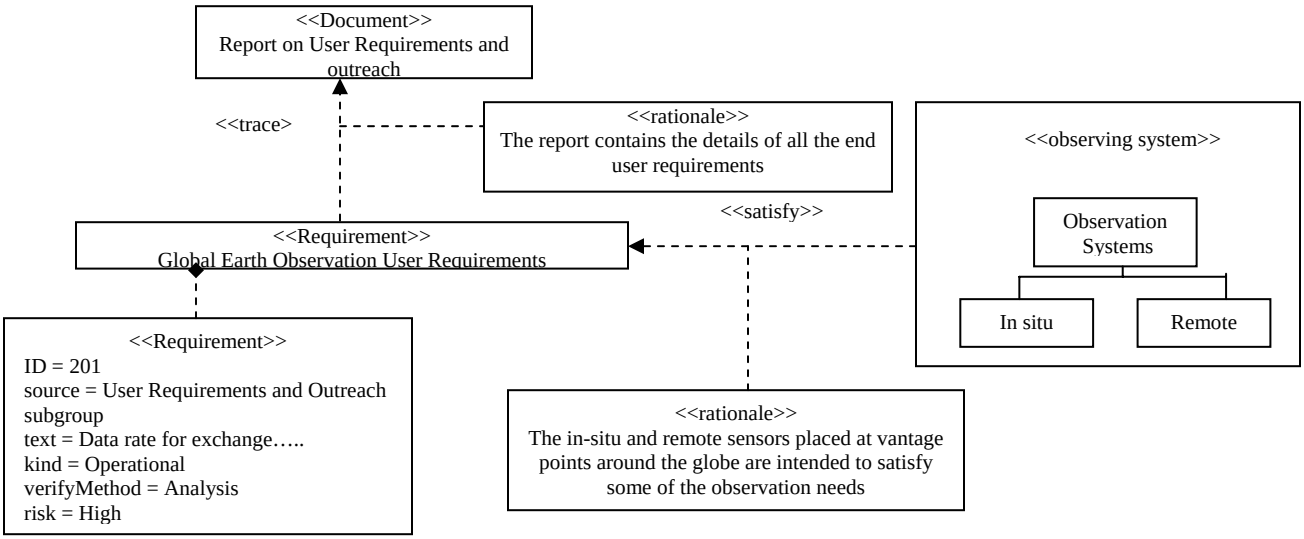

Figure 2. Requirements Diagram (II) of the GEOSS

Activity Diagrams

The 'Gather Tsunami Data/Information' use case can be further elaborated using the activity diagram. The diagram demonstrates how SysML allows modeling at the operational level. The activity diagram helps us in knowing the flow of information, data or material from one system

to the other in net-centric systems of systems environment. Swimlanes allow modelers to group activities together thus helping in assigning responsibilities. In SysML controls can also disable actions. SysML also allows modelers to specify both continuous and discrete flows with the option of defining the rate at which the entities flow. Figures 6, 7 and 8 show the activity diagram for the 'Gather Tsunami Data/Information' operation.

Description:
An extreme **undersea earthquake** is *recorded* by the **Global Seismographic Network**, one of the systems participating in **GEOSS**. **The worldwide system of regional tsunami warning centers** will *send* an **early warning** to *pre-designated authorities in Indonesia*. To confirm if the quake had generated a Tsunami, **seismic data** would be further *refined and combined* with **data from coastal tide gauges and buoys giving deep-ocean sea level**. Based on numerical methods, a **tsunami forecast** could be *prepared* for areas not yet affected. That information, along with **probable tsunami arrival times**, would be sent to the same authorities. If a Tsunami threat exists, **tsunami hazard zonation maps** are *prepared* by the **data processing node** showing areas vulnerable to tsunami run-up, areas of safety and evacuation routes. The required **high-resolution shoreline topography** and **near-shore bathymetry** are obtained from the **data producing node**. The **national emergency managers** will *work* with the **regional centers** to *facilitate rapid data exchange and co-ordination* of warning information. The national emergency managers will *send an alert* signal as **voice on radio and telephones, text captions on television, messages on highway signs, or signals for sirens**. **People and emergency teams** would *focus on relief efforts*. **Observation satellites** are then re-tasked to *image the likely affected coastal areas.*

Figure 3. 'Gather Tsunami Data/information' Use Case Description

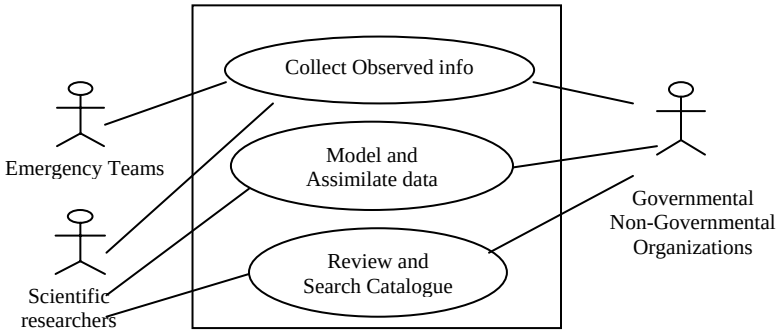

Figure 4. Use Case Diagram

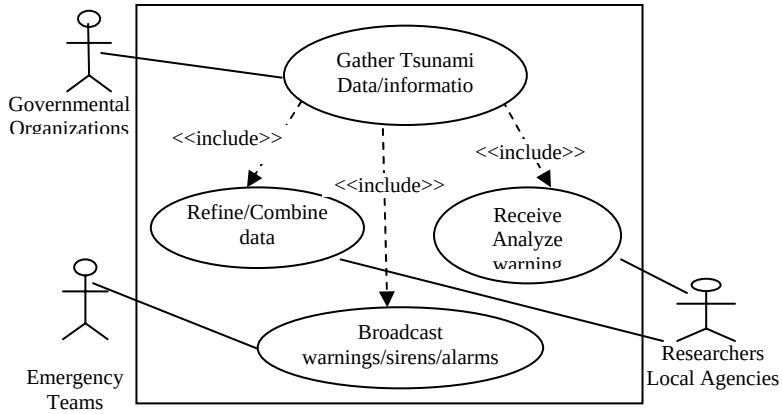

Figure 5. Use Case Diagram

In Figure 6, the Global seismographic network, World-wide tsunami warning centers, Ocean observation system are all grouped under the observing node. These three systems are responsible for continuous observation of parameters which are potential indicators of Tsunami. The Global seismographic network records an earthquake and this is represented as an action. The Global seismographic network then sends the 'raw seismic data' to the data processing node and also sends the 'Intensity' information to the 'Worldwide Tsunami Warning Centers'. The 'raw seismic data' and 'Intensity' are modeled as objects that are exchanged between the systems. The exchange of information is <<discrete>> in both the cases and is modeled by indicating as shown on the paths. The off page connectors have been used to connect flows across the multiple diagrams.

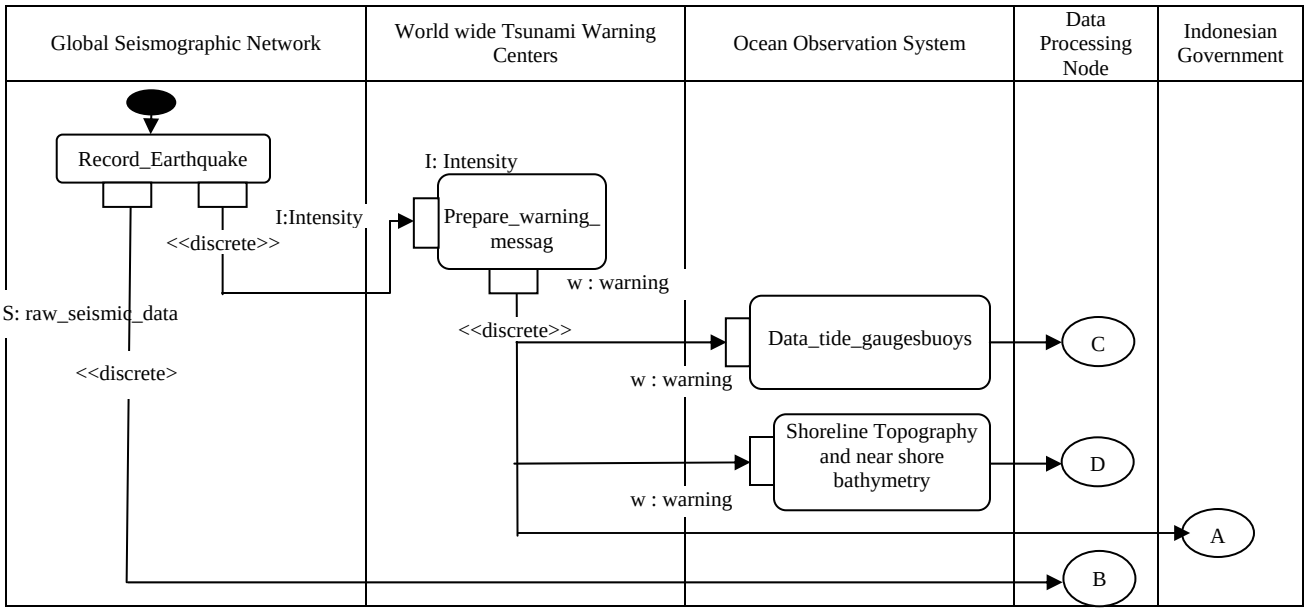

Figure 6. 'Gather Tsunami Data/Information' Activity Diagram – 1

In Figure 7, 'SendSignalEvent' has been used to indicate that a Tsunami warning message has been sent by the observing node to the government. In Figure 8, after the Government obtains the shoreline data and the zonation maps from the data processing node, a review is conducted with the information at hand in order to inform the civil society about the earthquake. This is modeled using the fork node. The review information is utilized to send alerts to the civil society through radio/telephone, television and the highway systems. Note that the alerts are considered to be <<continuous>> as indicated in the diagram.

Sequence Diagrams

Each of the activity diagrams are time-sequenced using the sequence diagrams. The sequence of operation and message/data exchanges that take place to accomplish each of the use case is shown using the activity and sequence diagrams. Sequence diagrams especially help in time ordering the communications among the block structures. The sequence diagram for the 'Receive/Combine data' use case is as shown in Figure 9. It can be seen that each of the system in the GEOSS is represented as blocks.

Block Definition diagrams

Block Definition Diagrams are analogous to class diagrams of UML. Figure 10 shows the system breakdown structure of the GEOSS. The composition symbol is used to show that the System of Systems is composed of the observing system, data processing/producing node and the network management system. Each system of the GEOSS is modeled as a block. SysML defines a stereotype of UML classes called <<block>>. The description of each of the block with the attributes and operations is provided in Figure 11. Class diagrams in UML are usually built from the collaboration diagrams as demonstrated in [Lee et al. 2003], which are built based on the interactions between the objects. The class diagram may not clearly outline the breakdown structure of the whole system. The messages or data that are exchanged between the various instances of the classes are represented as non-static classes in UML but in SysML all entities that flow into and out of the block are represented using FlowPorts and ServicePorts.

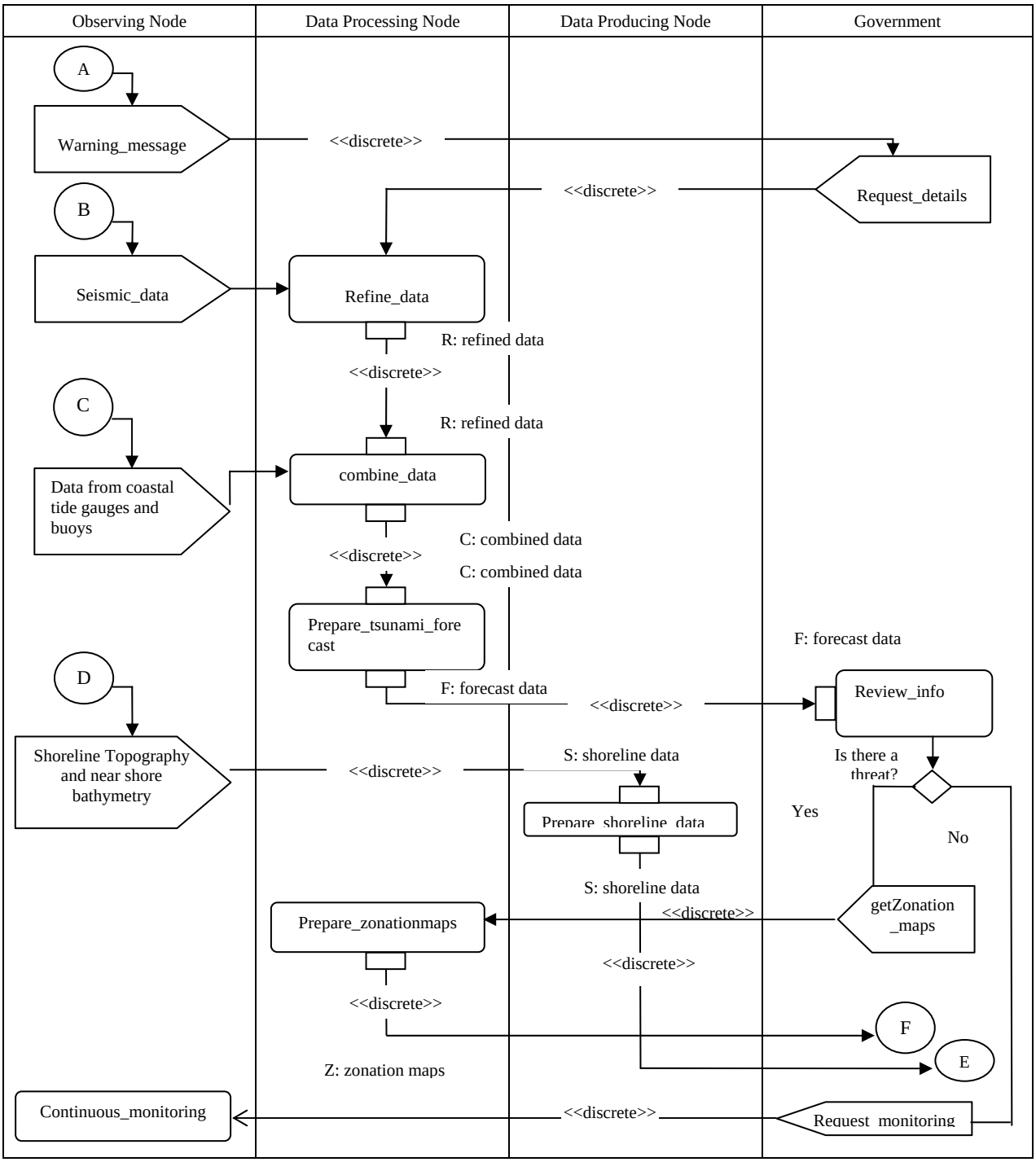

Figure 7. 'Gather Tsunami Data/Information' Activity Diagram – 2

In Figure 11, the block definition diagram of the observing node is shown with the attributes and operations. Attributes capture information about the state of the transmission. All the attributes have been typed using a user defined Value Type - 'data'. All the inputs and outputs that flow between the system and its environment are specified using the FlowPort. For Ex: 'Warning' and 'raw seismic data' are shown to be inputs to the observing Node. As outlined in the activity and sequence diagrams the data processing node provides zonation maps to the government upon request.

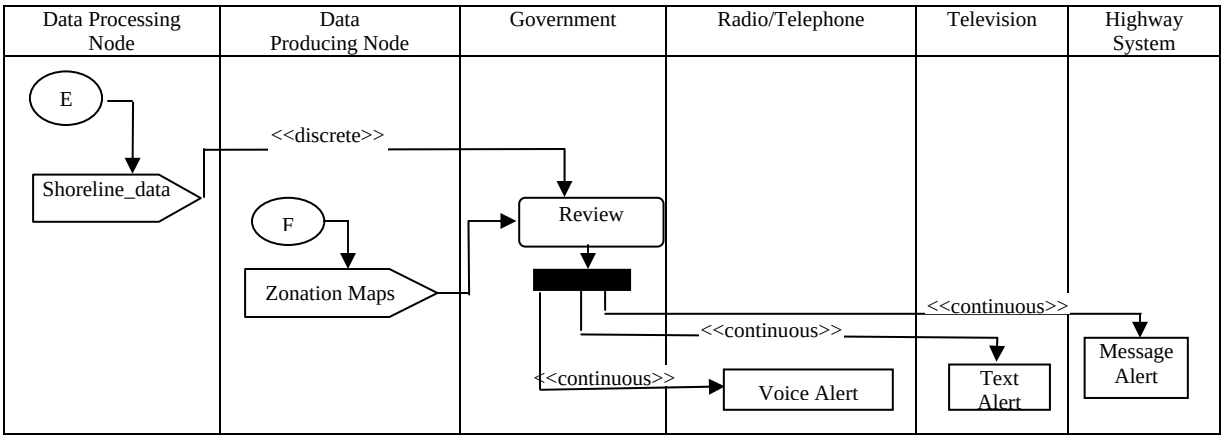

Figure 8. 'Gather Tsunami Data/Information' Activity Diagram – 3

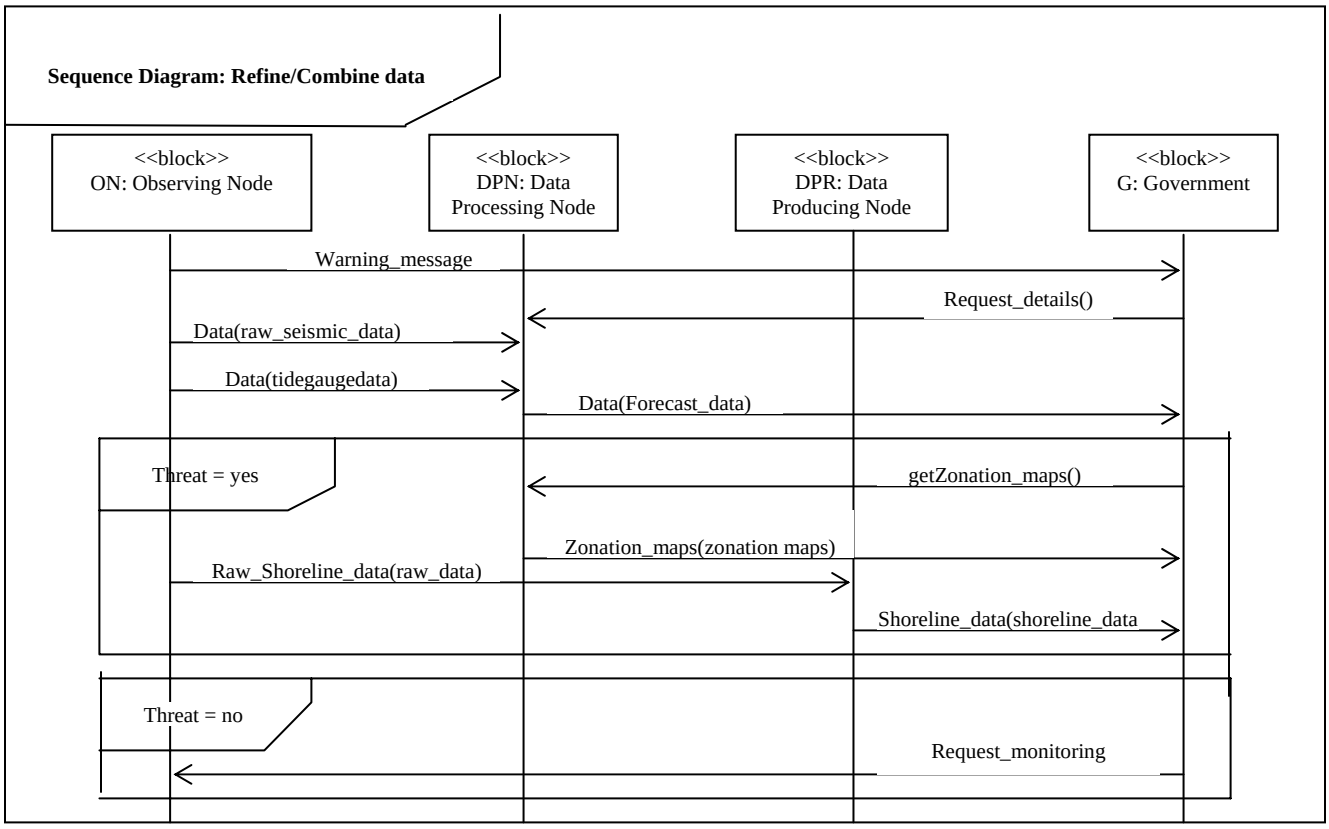

Figure 9. Sequence diagram for the 'Refine/Combine data' use case

These services are represented by the interfaces 'iGovtDet' and 'iDpnData' and are offered to the government through the service port on the data processing block. The operations to the various blocks were allocated using the SysML allocations (not shown here). More information about allocations can be found from [SysML, 2005]. The data exchange between the various nodes/systems is not limited to electronic networks; data/messages could either be exchanged through satellites or by floppy disks sent by mail from remote locations.

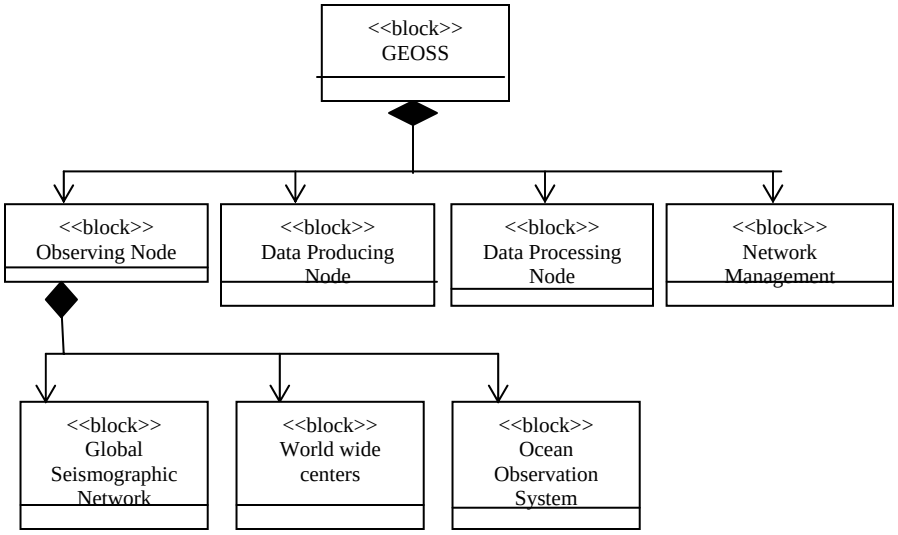

Figure 10. System breakdown structure of the GEOSS

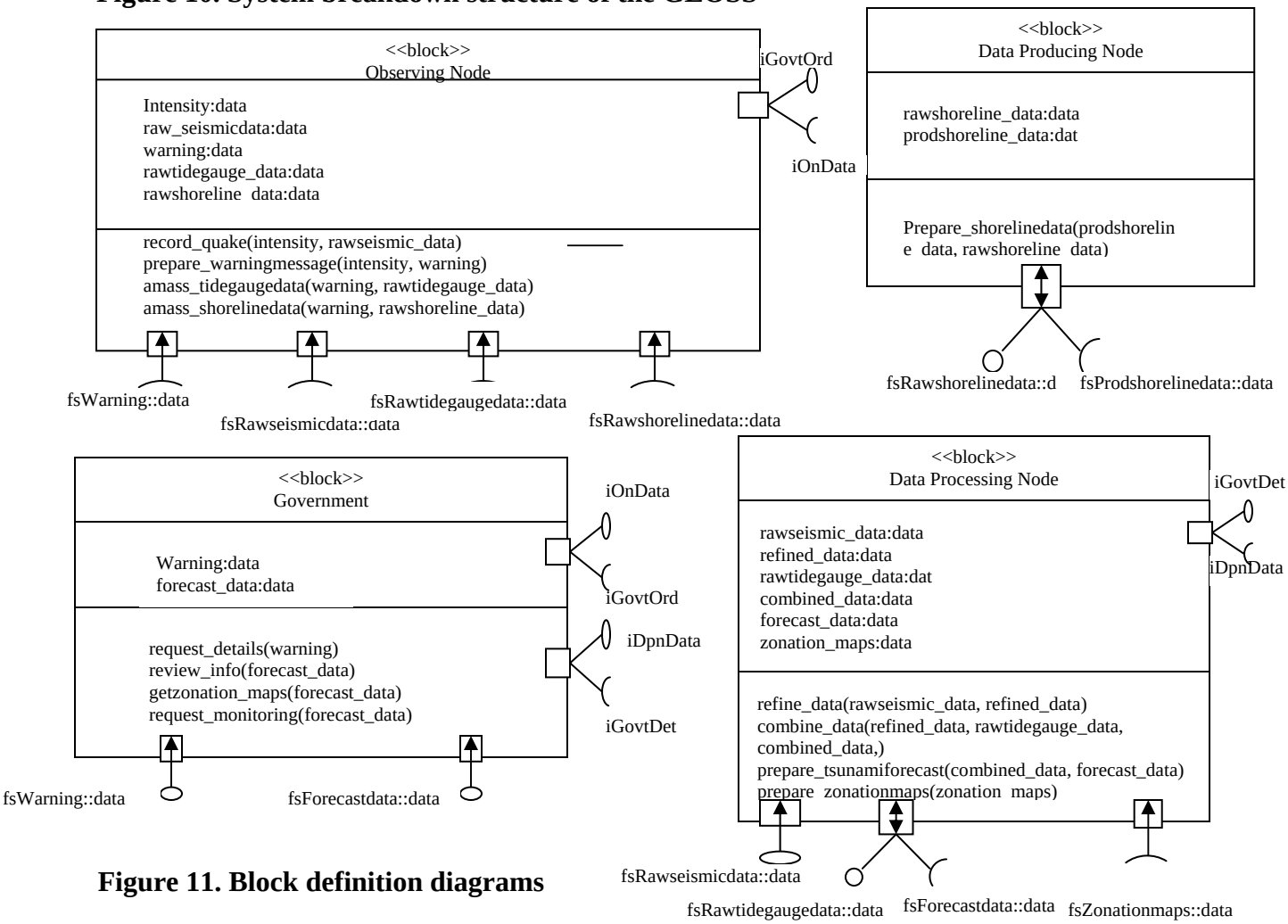

Figure 11. Block definition diagrams

Conclusions and Future Work

This paper demonstrates the use of SysML to model Network Centric System of Systems. The paper tries to demonstrate some of the advantages of using SysML to model SOS as compared to UML (See Table 1). The advantages of using SysML can be seen with the use of SysML Block Definition Diagrams and SysML Requirements Diagrams.

Both these diagrams have something unique to offer. SysML allocations are very helpful to build the Block Definition Diagrams. However, the advantages of SysML have to be further demonstrated at different levels of abstraction.

Table 1: UML Vs SysML

UML	SysML
Requirements are usually modeled using Use Case diagrams	Requirements can be modeled using Requirements diagram
Only discrete flows can be modeled using activity diagrams	Both Discrete and Continuous flows can be modeled using activity diagrams
Controls can only enable actions in activity diagrams	Controls can both enable and disable actions in activity diagrams
There is no way to specify the rate of flow of entities between systems	Rate of flow of entities between systems can be specified in the activity diagrams using the <<rate>> stereotype
Classes are used to develop the static model of a system	A stereotype of classes called <<block>>is used to develop the static model of a system
Messages/data are represented using non-static classes in UML	All messages/data that flow into and out of the block are represented using Flow ports and Service ports

Also, from a modeler's perspective validating the model thus built is very important. Once SysML diagrams are developed they can be converted to colored petrinetable form for model evaluation. Colored petrinets (CPN), is a graphical oriented language for design, specification, simulation and verification of systems. [Jensen, 1998] provides some information about CPN. [Lee et al. 2003] demonstrates that colored petrinets can be used to simulate and validate the models built using UML. A similar attempt for systems modeled using SysML will be of interest to the modeling community.

References

Daniels, Jesse, and Terry, Bahill, "The Hybrid Process That Combines Traditional Requirements and Use Cases." Systems Engineering, Vol. 7, No. 4, *Wiley Periodicals, Inc. 2004*

Global Earth Observation System of Systems, "10-Year Implementation Plan Reference Document", http://earthobservations.org/, 2005.

Hause, Matthew, and Thom, Francis, "Modelling High Level Requirements in UML/SysML." presented at INCOSE Symposium 2005, Rochester, USA.

Hause, C., Matthew, Thom, Francis and Moore, Alan, "Model Driven Systems Engineering – More Than Just Pictures." The International Council on Systems Engineering, Mid Atlantic Regional Conference, Nov 2 - 4, 2004.

Hause, C., Matthew, and Thom, Francis, "Systems Engineering with Solution Spaces." The International Council on Systems Engineering, 14th Annual International Symposium 2004, June 20 – 24, Pierre Baudis Congress Centre, Toulouse, France

Hause, C., Matthew, "Rebuilding the Tower of Babel – The Case for UML with Real-time Extensions." INCOSE Spring Symposium 2001, May 14 - 16, INCOSE UK chapter.

Jensen, Kurt, "An Introduction to the Practical Use of Coloured Petri Nets". In: W. Reisig and G. Rozenberg (eds.): Lectures on Petri Nets II: Applications, Lecture Notes in Computer Science vol. 1492, Springer-Verlag 1998, 237-292

Holt, Jon, "UML for Systems Engineering: watching the wheels." Second Edition

Lee, W., Wagenhals, Sajjad, Haider, and Alexander, H., Levis, "Synthesizing Executable Models of Object Oriented Architectures." *Systems Engineering Journal*, Vol. 6, No. 4, 2003 Wiley Periodicals, Inc.

Lee, W., Wagenhals, Insub, Shin, Daesik, Kim, and Alexander, H., Levis, "C4ISR Architectures: II. A Structured Analysis Approach for Architecture Design." *Systems Engineering Journal*, Vol. 3, No. 4, 2000 John Wiley & Sons Inc.

SysML, "Systems Modeling Language (SysML) Specification." version 1.0 alpha, 2005.