Hail Suppression Information System Development Using Unified Process

Dejan D. Rančić

Abstract - **The paper describes Hail Suppression Information System development using special version of the unified process named Rational Unified Process (RUP). This methodology is chosen because of the great supporting software tools made by Rational company. The new hail suppression information system should automate entire hail suppression activities and increase precision and efficiency of the entire system. This should be done avoiding major changes in the existing methodology. The system is based on PC technology enhanced with additional hardware for the real-time radar signal processing. For the system model description according to RUP methodology, Unified Modeling Language (UML) notation is used.**

Keywords –**RUP, UML, Radar, Visualization, Hail suppression**

I. INTRODUCTION

Information system for hail suppression has been developing for last few years through cooperation of Serbian Hail Suppression Service and Electronic faculty of Niš [1]. The system should automate hail suppression activities and increase efficiency of the entire process. At the same time, the existing hail suppression methodology should stay almost unchanged. At the end of the research period, completely functional prototype is developed and has been installed at three radar sites in Serbia for testing. The usage of such system was enabled precise requirements definition and stable architecture design of the entire system. Because of great system complexity, we chose RUP methodology for the final system development. RUP methodology became industrial standard for the software development in last few years. Brief description of the methodology as well as its appliance to the hail suppression information system development is presented in this paper. Developed model from beginning phases of the methodology is also described. For the model description, the UML notation is used. UML also became industrial standard for the software model description if object-oriented methodology is used.

II. RUP

RUP is the process framework developed by Rational Software Company. It is an incremental and interactive development methodology based on industry-proven best practices. The goal of methodology appliance is successful system deployment to the user. Over time, a RUP-based project goes through four phases [2]: Inception, Elaboration, Construction and Transition.

Each phase contains one or more iterations and each phase (except Inception phase, of course) should result by executable version of the system, which can be used for the testing. Of course, this methodology is not a rule and can be customized to fit needs for almost any project.

III. UML

UML is Unified Modeling Language for specification, visualization, design and documentation of software elements [3]. It is fusion of OMT (Object Modeling Technique) methodology, Booch's methodology, and OOSE (Object Oriented Software Engineering) methodology that are developed independently. The idea was unifying object-oriented methodologies as well as notation for object-oriented technologies. UML includes the best concepts from data modeling, process modeling, object-oriented modeling and component modeling.

UML can be used with many methodologies for software development, but it gives the best results if RUP methodology is used. Because of that, we chose RUP and UML combination for the Hail information system development. UML uses different types of diagrams for the static and dynamic system features describing. These are [3]:

1. Use-case diagrams,
2. Class diagrams,
3. Sequence diagrams,
4. Collaboration diagrams,
5. State diagrams,
6. Activity diagrams,
7. Component diagrams, and
8. Deployment diagrams.

Of course, it is not mandatory to use all types of diagrams during system modeling, especially from the reason that some diagrams are isomorphic (one diagram can be transformed to another – sequence diagram and collaboration diagram, for example). Detailed description of UML diagrams is out of scope of this paper, and because of that we will describe only their appliance during hail suppression system development.

Dejan D. Rančić is with Faculty of Electronic Engineering,
Beogradska 14, 18000 Niš, Yugoslavia,
E-mail: ranca@elfak.ni.ac.yu

IV. HAIL SUPPRESSION INFORMATION SYSTEM MODELING

Twelve meteorological radar centers and 1600 hail suppression stations (HSS), adequately distributed all over Serbia, represent the framework of the hail suppression system in Republic Serbia. Command of the system and coordination in hail suppression activities are performed by operational-methodological center in Republic Hydrometeorological Service in Belgrade. The primary goal of the system is to reduce hail damage to agricultural crops in Serbia, using cloud seeding method by rockets. Basic purpose of the radar center is to detect cumulonimbus clouds with high probability, to perform accurate measurements of all relevant parameters of hail cells in clouds, to determine launching elements (azimuth, elevation and timing) for rockets carrying seeding material, to determine suitable hail suppression stations with great degree of efficiency and command them. The existing methodology on radar centers (without new system installed) involves at least three operators responsible for separate tasks [4]. The first operator works by the radar and he is responsible for cloud monitoring and tracking. Manual cloud parameter extraction provides a hail danger estimates. The radar operator determines if the cloud should be seeded. When such a decision is made, the radar operator transfers all necessary parameters (the approximate geographical coordinates, size, estimated speed and direction and type of the cloud) to the second operator working on the map board. The map board operator draws the circle-like model of the cloud on the map and using rocket trajectory data tables determines which HSS will be activated. The map board operator also asks for launching permission from the Air Traffic Control Center and, after giving permission passes the launching parameters to the third operator that is responsible for communication with the HSS. The HSS communication operator then issues launching commands to each chosen HSS.

Introducing hail suppression information system we expect to achieve faster and more precise seeding process and, as a result, an increased efficiency. The described methodology represents a part of experts knowledge used for building an information system supporting the cloud seeding. The idea about phases and actions that must be supported by computers was formed and implemented. All parts of old methodology should stay but improved in terms of speed and exactness. Some steps were radically changed in methodology and new products were added.

Fig. 1 shows the basic architecture of the prototype that was made for the hail suppression system. The basic components are [4]:

- Non-Doppler MITSUBISHI RC34A weather radar,
- Main workstation,
- Two workstations for communication with the command center and hail suppression stations, and
- Intelligent microwave link for future weather radar networking.

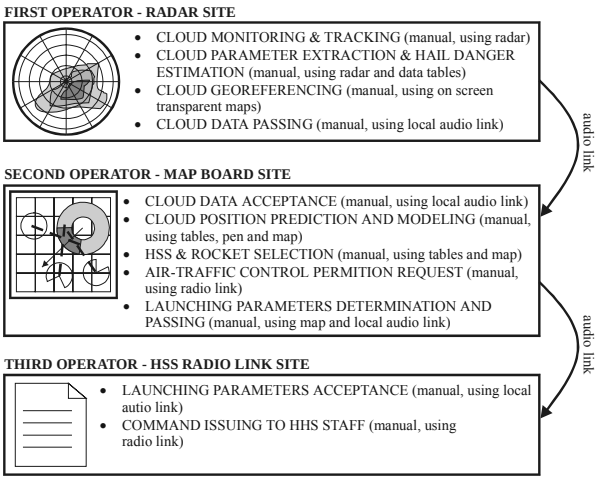

Fig. 1. Prototype architecture

According to the RUP methodology, developed prototype is considered as outcome of one iteration in software development. This prototype helps us to refine software requirements. Further, we continue with system modeling. According to RUP templates we use the "4+1" view model [5] (see Fig. 2).

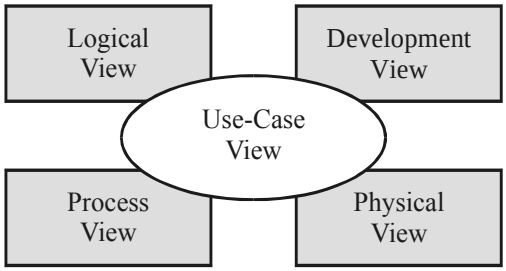

Fig. 2. The "4+1" view model

This model consists of five different views: Use-case view, Logical view, Process view, Development view, and Physical view.

Logical view describes static features of the system, while other views describe dynamic features.

V. USE-CASE VIEW

This view uses use-case diagrams and sequence diagrams to describe system from users point of view. The use-cases are in some sense an abstraction of the most important requirements. This view is redundant with the other ones (hence the "+1"), but it serves two main purposes:

• as a driver to discover the architectural elements during the architecture design, and

• as a validation and illustration role after this architecture design is complete, both on paper and as the starting point for the tests of an architectural prototype.

Fig. 3 shows one use-case diagram from the system as illustration.

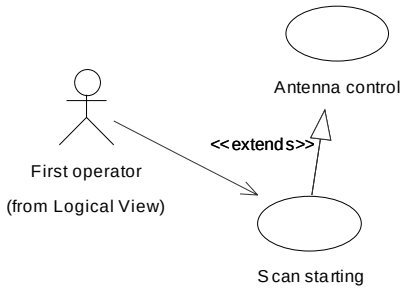

Fig. 3. Use-case diagram

V. Logical View

The logical architecture describes most important classes in the system and their organization into packages and subsystems as well as packages and subsystems organization into layers. According to the RUP template, we use modification of three-tier software architecture. Applying this model our system has been decomposed into four layers: **User interface**, **Services**, **Problem domain**, and **Data management**. The layer **User Interface** consists of classes for the user-system communication and control. Classes from this layer use those from **Problem domain** layer and **Services** layer. In fact, classes from **User interface** layer transform user commands to methods execution from other layers. The **Services** layer consists of control classes, which determine system behavior. Classes from this layer use those from **Problem domain** layer and **Data management** layer. The **Problem domain** layer consists of classes, which represent key abstractions from the problem domain (clouds, rockets, radar, etc.). Classes from this layer use classes from **Data management** layer. The **Data management** layer consists of classes for data storing and retrieve to/from database and files.

The appropriate class diagram describes each layer. Fig. 4 shows the part of the **Problem domain** layer as illustration.

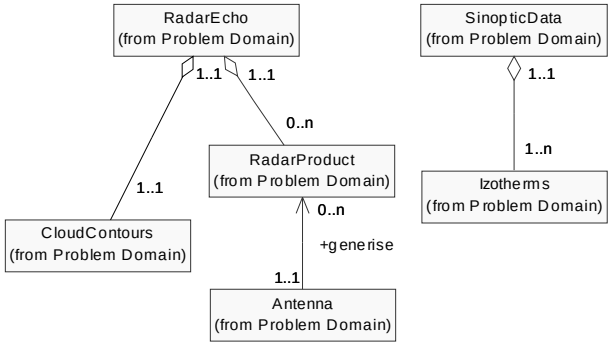

Fig. 4. Class diagram from the **Problem domain**

Each class is described by their methods and attributes. Fig. shows class **CloudContours** from the Problem domain as illustration.

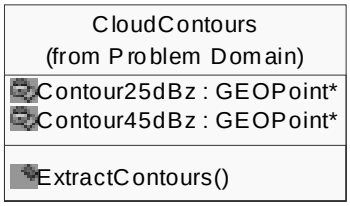

Fig. 5. CloudContour class from the Problem domain

VI. Process View

The process architecture describes most important processes and threads in the system and their organization. Processes are executing using independent address area while threads use address area of the existing process. Hail suppression information system consists of following processes and threads (Fig. 6): **MWSApp** (process), **WS1App** (process), **WS2App** (process), **ProductGeneration** (thread), **Acquisition** (process), and **CloudSeeding** (thread).

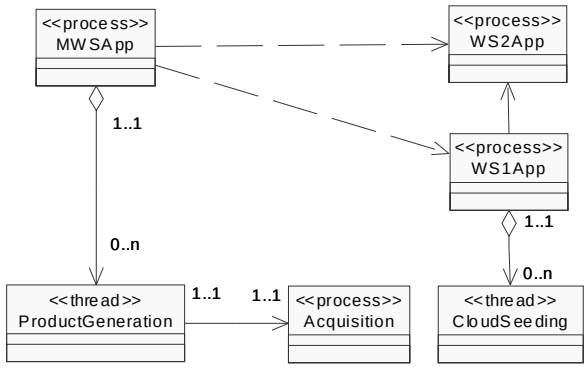

Fig. 6. Process View

The process **MWSApp** is the main workstation application and this process controls main workstation and radar. This process can initiate **ProductGeneration** thread. Thanks to the fact that **ProductGeneration** is thread, user can perform other actions (parameters measurements, for example) while thread is running.

The process **WS1App** is the first workstation application and this process controls cloud seeding functions. This process can initiate **CloudSeeding** thread. As in the previous case, user can perform seeding operation to multiple clouds in parallel.

The process **WS2App** is the second workstation application and this process controls communication with the Command Center and HSS. The **Acquisition** process is running on the DSP and it controls data acquisition process.

The thread **ProductGeneration** is running in the address area of the **MWSApp** process and it controls radar products generation (PPI, RHI, CAPPI,…).

The thread **CloudSeeding** is running in the address area of the **WS1App** process and it controls cloud seeding functions.

VII. Development View

The development architecture focuses on the actual software module organization. Basic implementation and components in hail suppression system are (Fig. 7): **Radar** (executable), **GISBoard** (executable), **HSSControl** (executable), **HASISDB** (database) and **SigProc** (executable).

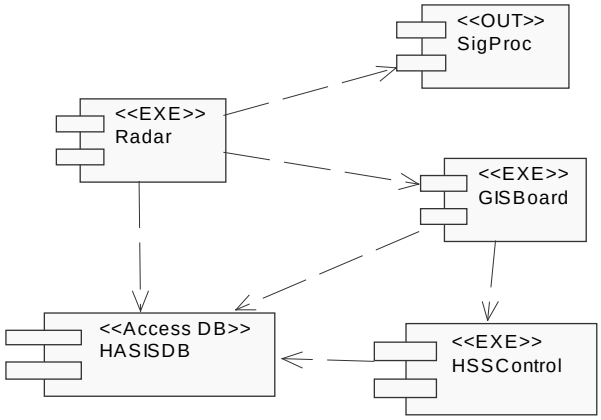

Fig. 7. Development View

The Radar component is executable deployed on the Main Workstation node (Fig. 8) and consists of **MWSApp** and **ProductGeneration** implementation.

The **SigProc** component is executable deployed on the **DSP** node and consists of **Acquisition** process implementation.

The **GISBoard** component is executable deployed on the First Workstation node and consists of **WSApp1** and **CloudSeeding** implementation.

The **HSSControl** component is executable deployed on the Second workstation node and consists of **WS2App** process implementation.

The **HASISDB** component is not executable. It is a **MS Access** database and it is deployed on the Second Workstation node. This database is used for data management.

VIII. Physical View

The physical view describes physical nodes in the system and their spatial deployment. Hail suppression information system has four nodes (Fig. 8): **Main workstation** (computer), **First workstation** (computer), **Second workstation** (computer), and **DSP** (additional hardware).

All workstation are connected in the Local Area Network (LAN), while DSP node is the part of the **Main workstation** and uses PCI bus for data transfer.

Fig. 8 shows deployment diagram of the system.

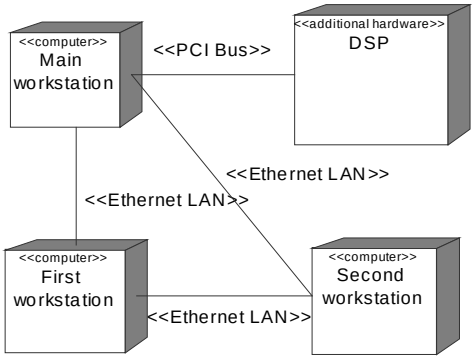

Fig. 8. Deployment Diagram

IX. Conclusions

The RUP methodology for the development of software intensive systems together with the UML notation became really powerful aims in the area of software development. They became very popular in recent time, and we can freely speak about them as industrial standards. Well documented, precise defined with existence of very powerful supporting software tools, tutors, and templates, often make RUP + UML combination as first pick in software development. This was the reason for choosing this combination for development of such complex system as hail suppression information system. The paper shortly describes basic steps in the system modeling and gives different UML diagrams that represent parts of the entire system as illustration. Because of the huge complexity of the system, it was impossible to describe system model in detail. Anyway, this short description of the system can be used as template for the modeling of similar systems and hence it can be very useful for those who chose RUP and UML for system development.

References

[1] Rančić, D., Smiljanić, M., Đorđević-Kajan, S., Kostić, A., Eferica, P., Vuković, P., Vučinić, Z., "Radar Data Processing for Cloud Seeding in Hail Suppression Information System", *Proceedings of the RADME 98 - Theoretical, Experimental and Operational Aspects of Radar meteorology*, Rome, Italy, pp. 137-149, 1997.

[2] Jacobson, I., Booch, G., Rumbaugh, J., "*The Unified Software Development Process"*, Addison-Wesley Publishing Company, 1999.

[3] Booch, G., Rumbaugh, J., Jacobson, I., "*The Unified Modeling Language User Guide"*, Addison-Wesley Publishing Company, 1999.

[4] Rančić, D., Smiljanić, M., Đorđević-Kajan, S., Kostić, A., Vuković, P., Eferica, P., Vučinić, Z., "Software Support for Cloud Seeding in Hail Suppression Information System", *Proceedings of the YU INFO 99*, Kopaonik, Yugoslavia 1999. (in Serbian)

[5] Kruchten, P., "Architectural Blueprints - The "4+1" Model of Software Architecture", *IEEE Software* 12(6), pp. 42-50, November 1995.