Development of an Interface to a Spacecraft Simulator Empowered by Virtual Reality

Christopher Shneider Cerqueira[1], Walter Abrahão dos Santos[2] and Ana Maria Ambrosio[1]
Engineering and Space Technology[1], Applied Computing Laboratory[2]
National Institute for Space Research - INPE
São José dos Campos - Brazil
christophercerqueira@gmail.com, walter.abrahao@lac.inpe.br, ana.ambrosio@inpe.br

***Abstract* — Due to its large territory, Brazil relies on space systems to perform a myriad of supporting activities. Space systems design requires strong modeling and simulations techniques for achieving high performance. This article describes the development of a goal-driven user interface (UI) for spacecraft distributed simulations using a service oriented paradigm and supporting different space missions. From usability problems with traditional simulators UIs, described by satellite subsystem design engineers, this work proposes three dimensional visualization, natural interaction techniques, virtual and augmented reality as well as interaction with touch-screen and gesture recognitions. This research uses an open source C/C++ toolkit, designed to provide interactivity, networking and scripting capabilities for simulator developers, this facilitates in providing specific input or output driver engines. Finally, some actual release pictures and information which includes single-touch interaction in the Smart Board Screen and some further developments are described for future work.**

***Index Terms* — Interaction, virtual reality, simulation, interface, computer graphics, distributed systems.**

I. INTRODUCTION

The space exploration generally drives research and applications to areas such as rocket propulsion, life support, new materials, reliable computer algorithms, autonomous operations, etc. The applications of space exploration include Space Observation, Earth navigation, Communication, Meteorology and Remote Sensing. All space data gathered provides resourceful information to our day-to-day life. In Brazil, space missions are mainly performed by the National Institute for Space Research (INPE) and they play a key role for the vast Brazilian territory in water, fishery, agricultural and deforestation monitoring as well as weather/climate data gathering from ground sensing data platforms or obtained from images taken by artificial satellites [1].

A space mission is divided into four segments [2]:

- Ground Segment - provides communication with Space Segment to answer user's requests with the information acquired;
- Space Segment - provides data to the Ground Segment, usually could be a satellite, probe, capsules, space telescopes and space shuttles;
- Launch Segment - places the Space Segment into the space, characterized by a rocket propelled artifact and;
- User Segment - receives and uses the acquired data, e.g. scientists, media, agricultural companies and government.

After a satellite launch, an interconnection of the segments operates like is shown in Fig.1. The Ground Segment interfaces to the User Segment and the Space Segment; it is responsible to provide reliable information to the user communities. After launch, all tasks such as control and operation of the satellites and management of the mission are handled by the Space Segment elements. The Ground Segment is subdivided into four elements [2]:

- Mission Center - handles mission concept, evaluation, analysis and Mission Exploration and Payload Data;
- Control Center -handles operation control, simulation, flight dynamics, data handling and distribution;
- Network - handles the interconnection of centers, stations and the spacecraft, using ground and space links and;
- Ground Stations - handles the communication link between the Ground Segment and the Space Segment.

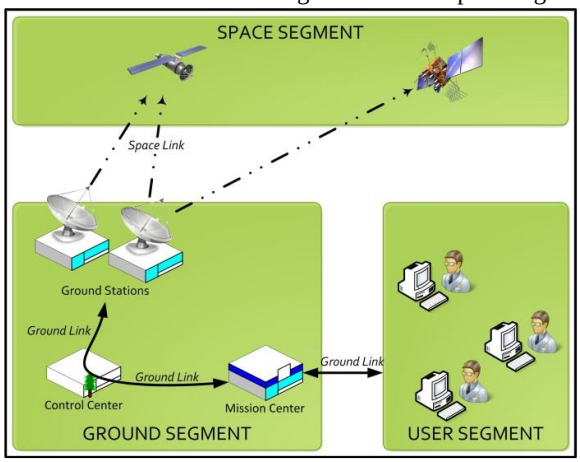

Fig. 1. Ground Segment and its relations to other space mission segments

Ground Segment provides facilities and resources to control and operate the Space Segment artifacts, as satellites. Activities of the Satellite Control Center, for instance, briefly depicted in Fig. 2, are divided into flight dynamics, mission planning from user requests, acquired data distribution to user and simulations

tasks. In this context, these simulations represent an important task to validate requirements and specifications, to train operators, to test solutions and to support public outreach of space research.

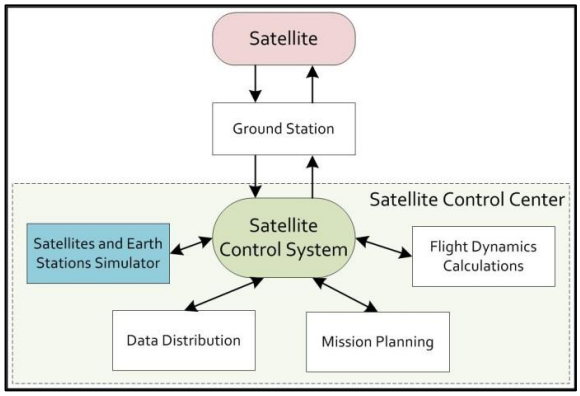

Fig. 2. Satellite Control Center Elements

In space programs, the stakeholders interested into operational spacecraft simulation look for different information and views as shown in Fig. 3. Therefore these interests must be understood before any development. For example, developers are mainly interested in model representations and user interactivity, managers want to validate operational scenarios, presenters and museums might want to exhibit the space mission to public and spacecraft subsystems´ engineers are interested in incorporating their hardware into a simulation loop [3], interconnect different operation models system budgets, check assembly and configurations. Nevertheless, most of the time, the simulator interface is not simple to provide fast information recovery, controllability, self-descriptiveness and, suitable user-friendliness [4].

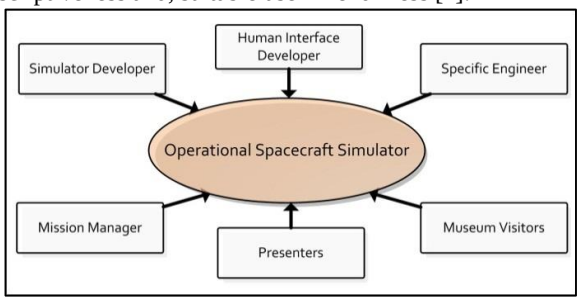

Fig. 3. Stakeholders diagram for an operational spacecraft simulator.

INPE has developed some simulators to its early missions. For the SCDs (Data Collecting Satellite from the Portuguese "*Satélite de Coleta de Dados*") the SIMS simulator started in 1991. Later, the SIMCS simulator was built to comply with the CBERS-1 (China-Brazil Earth Resource Satellite), used to support operators' training activities and operations details, such as attitude and orbit control telecommands. The FBMSIM simulator was designed to support operations team training for the French-Brazilian scientific micro-satellite [5] where one of the challenges consisted into specifying how the satellite behavior is represented in the final software product. It is hard to find a common language, understandable and acceptable by every stakeholder, and yet powerful enough to describe the satellite behavior [6]. Finally, as shown in Fig. 4 and 5, current developments use files, tables and graphics to represent the simulated information [7].

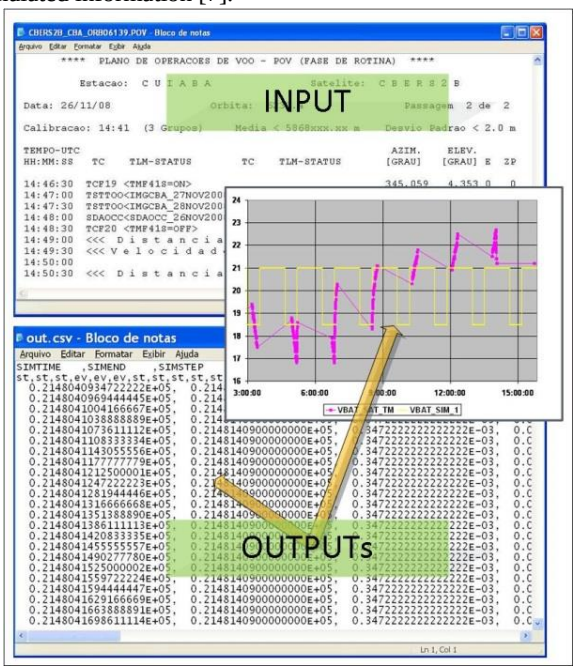

Fig. 4. Files and tables artifacts as CBERS simulation input/outputs.

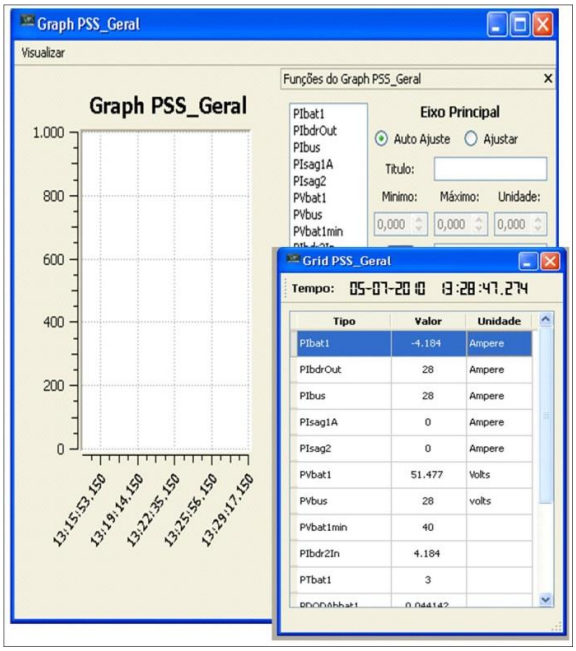

Fig. 5. Visual simulation outputs of a Satellite Power Subsystem shown as graphs and tables.

Computer Graphics (CG) and Computer Vision (CV) may provide alternative views and interactivity with simulation objects and the environment, allowing the development of several tools, training and, operations. Some of these

alternative interfaces can be created using Virtual Reality (VR) or Augmented Reality (AR) techniques [8]. An example of a VR application is the one developed by NASA (National Aeronautics and Space Administration) [9] to simulate the Mars' rovers with real pictures taken by the Pathfinder mission [10] as shown at Fig. 6. Similarly, Fig. 7a shows an example of AR initiative developed by ESA (European Space Agency) [11], where the system provides "*Just in Time*" and "*Just in Place*" information to ISS (International Space Station) astronauts helping in their operations. Finally, an initiative from NASA to upgrade the spacesuit HUD (Head-Up Display) with an AR layer can be seen at Fig. 7b.

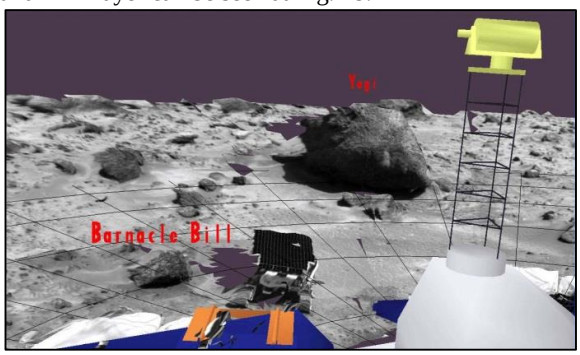

Fig. 6. Virtual Reality Simulation for a Mars' Rover [9].

Fig. 7. (a) ESA Augmented Reality system to ISS crew [11]. (b) NASA upgrades to HUD for spacesuits [9].

A goal-driven or object-driven simulation operation is based on goals defined by the simulation operator. This approach is different from the traditional setup of multiple steps or script development very common in gaming where it´s necessary to develop intelligent agents to interact with users [12]. Instead, given a goal, a planner elicits its requirements and negotiates with the environment in order to specify the necessary actions towards the desired result [13]. An example of a goal-driven technique is touching or clicking a globe to select the imaging area and the simulator generates the proper telecommands required to place the satellite on-board camera for image recording.

In this work, we architect a simulator interface using VR elements to help satellite subsystems´ engineers to address decisions regarding satellite fault and operational conditions, attempting to make the simulator more usable and responsive by fast information recovery to the user needs. The virtual 3D operation simulator shows the simulated spacecraft orbit and some simulation parameters. This allows interaction and variation of effects, based on visually goal-driven definitions instead of traditional programming scripts and/or windows point and click menus.

II. Related Work in Space Simulators

Simulators are key tools to Space Mission as they help the team work. Space Missions have usually seven phases [2] and at each one, simulators may be applied for specific tasks as shown Fig. 8 [15]. Simulators explore the requirements from different approaches and depths. At phase 0, usually are used mission analysis simulators as the STK (Satellite Tool Kit) [14], the following phases requires more specific and refined simulators to validate and optimize design parameters. At phase D, simulators may check systems and help on the spacecraft assembly. At operation, simulators may train operators on routine, degraded and emergency situations and help to evaluate and propagate orbits, planning maneuvers, check equipment conditions, etc. At disposal, simulators can help on reentrance calculations to choose the best point to start maneuvers, as well as the debris placement.

Some space simulators have virtual representations, some examples are shown in Fig. 9-11, respectively the STK[14], Orbiter [16] and the Celestia [17].

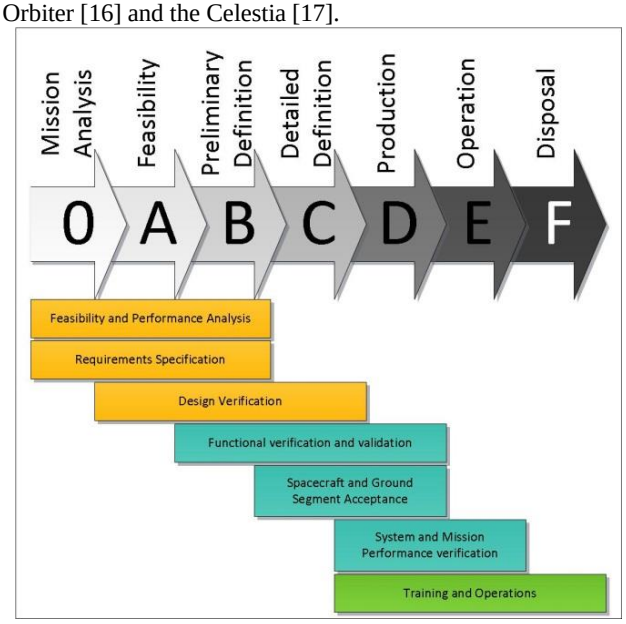

Fig. 8. Simulators are used throughout the Space System Lifecycle [15].

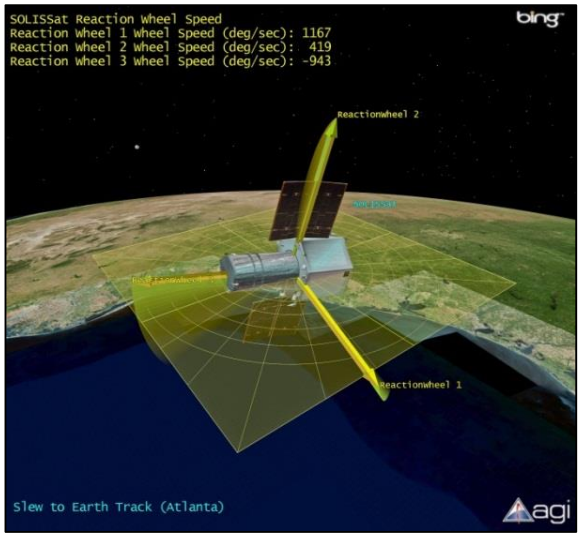

Fig. 9. Satellite Tool Kit (STK) [14].

Fig. 10. Orbiter showing a star sensor view [16].

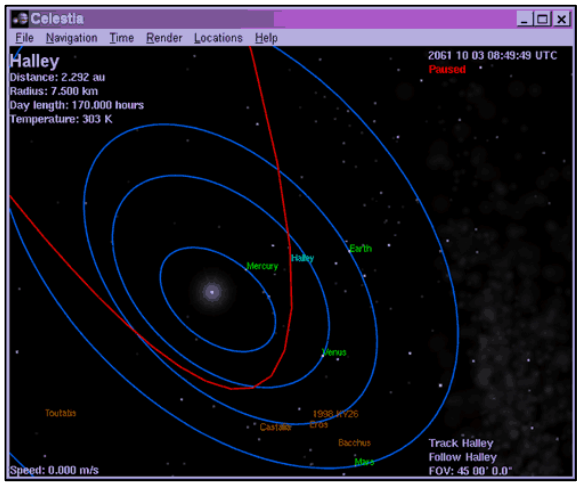

Fig. 11. Celestia Halley Comet 3D Simulation [17].

III. Decoupling Interface from Simulator Core

In a spacecraft design program, each specialized team develops subsystems and provides its parameters to simulation which are encapsulated into simulation models. Usually this software executes and controls all models and handles the Graphical User Interface (GUI) in a single software product.

However, as the model sophistication grows it demands more computational power, which can be solved with a distributed solution [18]. As shown in Fig. 12, each subsystem is modeled and simulated into smaller simulation cores, and they communicate with other subsystems and the main simulation control. A Simulation Control Kernel manages the simulation distribution tasks.

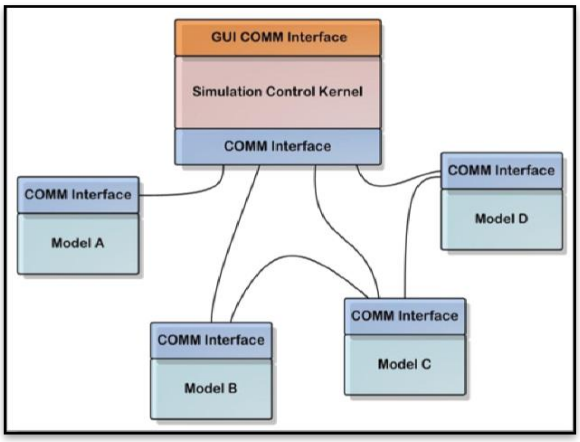

Fig. 12. A generic scheme for a distributed simulation distribution.

This approach allows specific engineers to develop their own models, verify and validate their functionality and link into a simulation chain.

Applying the decoupling concept to the GUI and the simulator, allows multiple users having their own interfaces for collaboration and simulation data visualization. More sophisticate interfaces can allocate more computer power into visualization and interaction strategies than in the intensive simulation itself.

Both, the Simulator Kernel (Model and Simulation Control) and GUI decoupling, can use the same communication provided by Service Oriented Architecture (SOA) [19].

A SOA implementation is usually achieved by web services. This technology is adequate for integration of low coupling heterogeneous systems [20]. The fundamental elements of a web service are shown in Fig. 13: the service consumer and provider and their relationship. The consumer requests a service by their description in the Web Service Definition Language (WSDL) to obtain the necessary information to consume the service. A broker finds the service, which is published using Universal Description Discovery & Integration (UDDI). The Simple Object Access Protocol (SOAP) is a standard protocol in SOA to interconnect systems and it uses XML (eXtensible Markup Language) metafiles to exchange data [21].

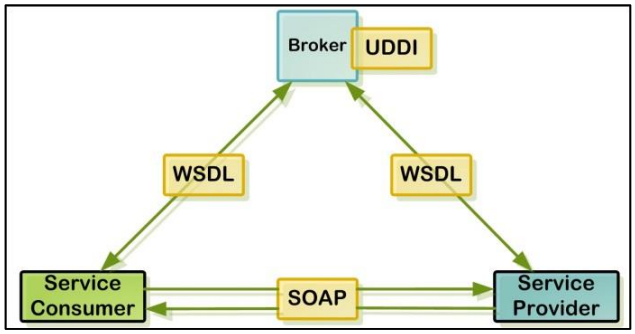

Fig. 13. Fundamental elements of a web-service.

SOA characteristics provide greater functionalities to services as reuse, low redundancy and maintainability. Those elements provide a way to decouple the GUI from the Simulation Core allowing the use of elements that demand intensive CV procedures, interaction state machines and complex rendering tasks.

IV. Simulator Interface Architecture

User interfaces evolves by advances in the available technologies for the interaction and data visualizations.

Input interaction allows the user to perform actions in the computational data. An Interaction Engine (IE) is responsible for understanding a group of possible user actions and generation of event handling to the system. The user actions come from different devices and methods: keyboard, mouse, touch-screen, augmented and gesture handling, voice or from a specific hardware devices.

An Interaction Engine with different input drivers (CV, Hardware and Voice) and some possible interaction sources is shown in Fig. 14. The drivers respond to behavior control structure that recognizes a meaning in the user action.

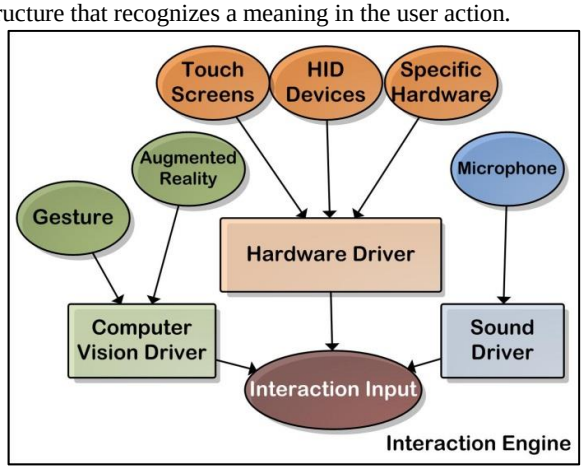

Fig. 14. The Interaction Engine and its drivers.

Output data is the feedback to user by the computer, environment as user or external event occurs. The output data can be visually and/or audible feedback to the Interaction Engine, so it provides more arguments to track meaning actions. Similarly, the Output Engine (OE) drives the virtual placement into monitors, projectors and/or HMDs (Head Mounted Displays) and the audio placement to speakers.

An Output Engine, with its output visual and sound drivers as well as possible output destinations is shown in Fig. 15.

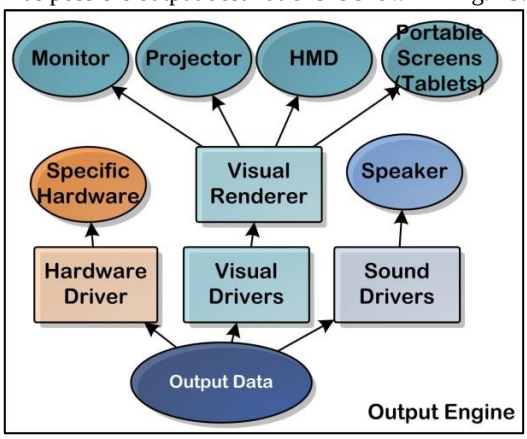

Fig. 15. The Output Engine and its drivers.

In order to interconnect an IE to an OE, a User Interface Core (UIC) can be used; it encapsulates groups of IE-OE creating the main user interface and interface widgets. The UIC is also responsible to treat external events from a Communication Link (CL). A macro view of the system is shown in Fig. 16, with multiple user transferring information to the IE; it is processed by the User Interface Core that has external signals from the CL, which is connected into a cloud infrastructure that provides OE data to user and back to the IE.

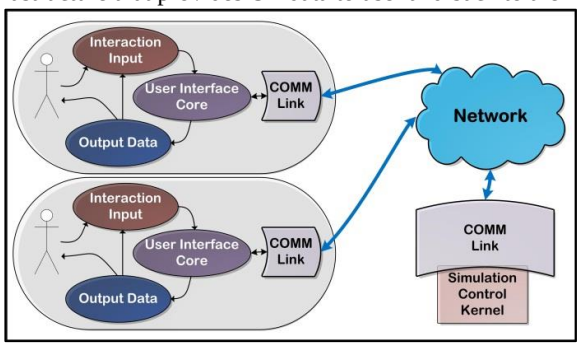

Fig. 16. System with multiple interfaces accessing a simulation.

In this work, the openFrameworks [22] has been chosen to address the IE, OE, UIC and CL implementations and it is an open source C++ toolkit designed to wrap together several commonly used library. This includes the following: graphics, audio input, output and analysis, video playback, grabbing and processing, utilities as database, file system tools, multithreading, network, logging and XML [23].

The openFrameworks architectural elements and its available resources are shown in Fig. 17.

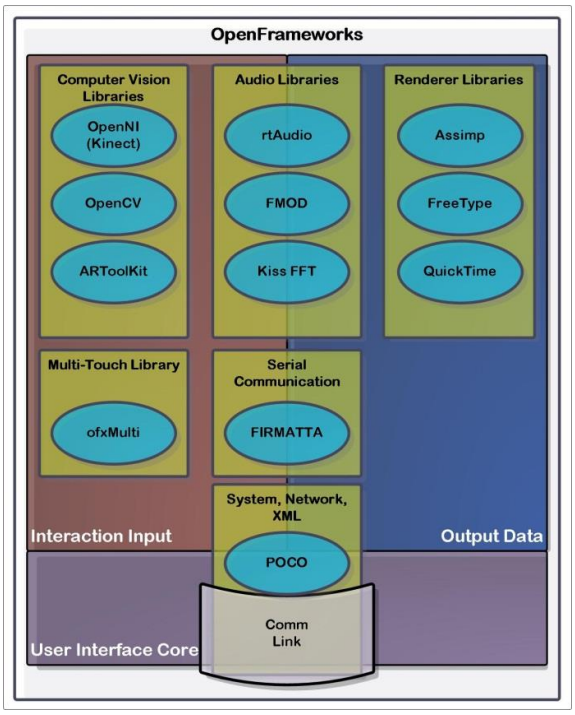

Fig. 17. openFramework and its Libraries.

The UIC is described by an XML file. The file describes the view setup, the objects properties (position, visibility, actions, etc.) and the objects behavior (click, drag, rotate, translate, change camera, etc). Most of the functions are hardcoded such as orbit placement and auto refreshment of external data. However this XML description provides flexibility to load widgets that calls other external data providers.

The complete architecture of possible plug-in elements for a spacecraft simulation interface that allows VR and AR is depicted in Fig. 18.

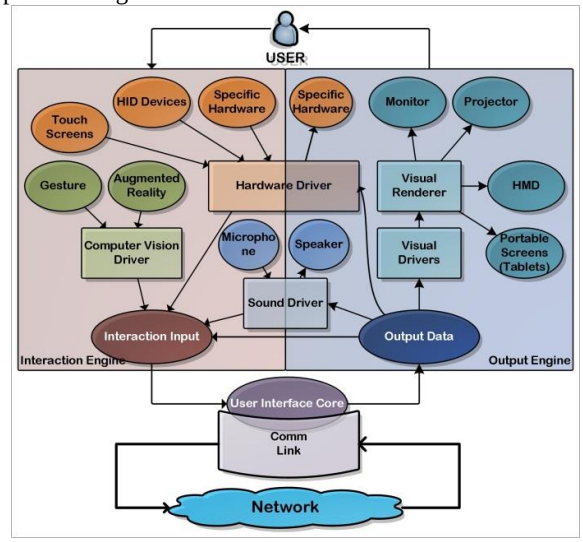

Fig. 18. Architectural elements for the visual interface to a distributed spacecraft simulation.

The connection of several interfaces into a simulation is illustrated in Fig. 19 where modules are distributed and connected to one another. The modules include an Environment module responsible to Earth magnetic field, and disturbance factors, a Ground Station module responsible for sending telecommands, visibility calculations and an interface to specific communication with other INPE software and the Satellite module broke-down into some modules (Flight Dynamics, Power Supply, Communication and On-board Computer).

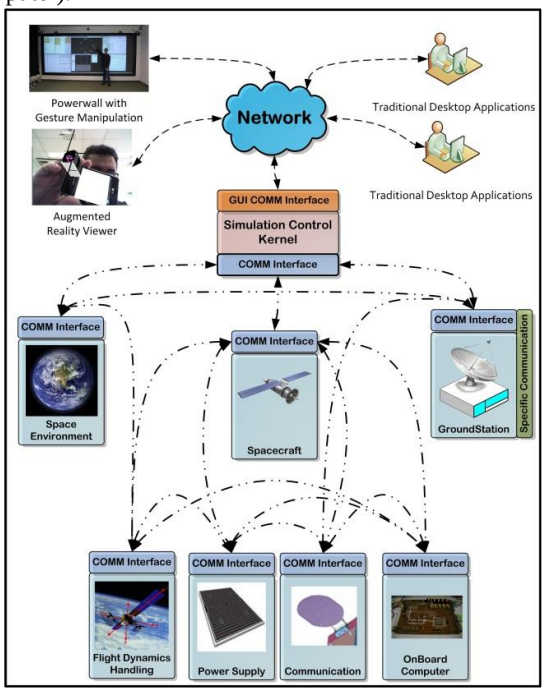

Fig. 19. Multiple interfaces connecting distributed simulation elements.

V. Simulator Development and Early Results

The simulator development is based in an evolutionary process of releases incrementally incorporating more functionality, starting with the first concepts and then evolving to aggregate re-configurability, communication and lastly, user adaptation and restrictions, namely:

1. Release 1: *Concept*
   a. Build basic screen.
   b. Apply mouse operation (Single Touch).
   c. Draw and change orbital elements in 3D.
   d. No external Communication.
2. Release 2: *XML, Widgets and AR*
   a. Re-evaluation of Release 1.
   b. XML file to build the screens and widgets behavior.
   c. Add AR view with behavior. [24]
3. Release 3: *Connection*
   a. Re-evaluation of Release 2.
   b. Multi-touch and OpenCV (Open Computer Vision) handling.
   c. Communication Services.
4. Release 4: *User*

a. Re-evaluation of Release 3
b. User constrains and access level.
c. File transferring.
d. Synchronization among interfaces.

At the point, the development is at first release and its current screen is described in Fig. 20 which shows the interface with a ground station visibility, and telecommands transfer timeline. Additionally, it also has a real time and simulated time indicator, the virtual 3D spacecraft and a 3D Three-Body System (Sun, Earth, and Satellite) view, and general tools.

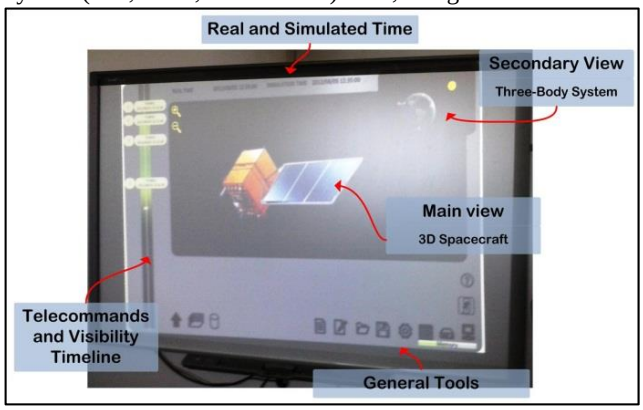

Fig. 20. Main screen first release with CBERS satellite in the background.

The current release (Release 1) has been tested with a Smart Board Screen [25] at the LABORARE (Educational Resources and Augmented Reality Laboratory) at UNIFEI (Itajubá Federal University). An example of a touch action when the user brings to first plane the 3D Three-Body System and touches into the spacecraft is demonstrated in Fig. 21 showing a menu to change the Keplerian Elements [26] [27][28].

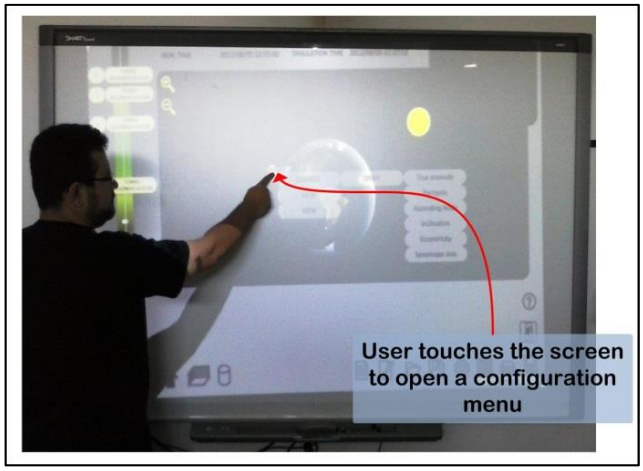

Fig. 21. Single-Touch interaction example for satellite orbital descriptions.

A change in the orbit inclination using a protractor symbol is shown in Fig. 22 helping the user to interact with the Smart Board Screen via multi-touch. Next steps for this research will include the virtual exploded view of the satellite using "touching" gestures capabilities to send telecommands or acquire telemetries from satellite equipment finishing Release 1 and starting further evaluations.

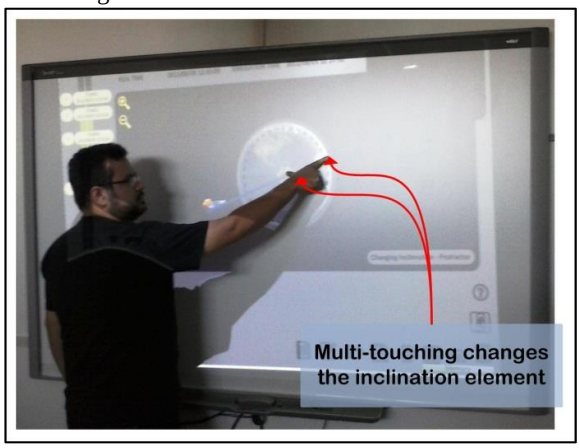

Fig. 22. Multi-Touch interaction to change a satellite orbit inclination.

VI. Conclusion

This work presented the early evolutionary development of a user interface to a spacecraft simulation using VR and later AR inclusion. This intends to create a common visual interpretation to the multiple stakeholders. The interface is designed to access data from an external source, in this case, from a spacecraft simulator.

The interface is empowered with natural manipulation and three-dimensional representations of data that contextualize the information into the all scenario instead of using fragmented date in files, tables and graphs.

A visual interface also allows goal-driven manipulation as the user can touch parts of the system and manipulate actions or plan future events.

Since service orientation is a technological solution to distributed data access, so this work also illustrates a possible design of a single simulator where its sub-modules are distributed and accessed through services. The decoupling of the simulator kernel and its modules can split the interface from the simulator. This allows the use of more specific interfaces, written in different languages and scenarios.

Future work will demand fulfillment of the development, usability testing and validation with the mission specific systems engineers. Finally, it fits also to a science museum where visitors can interact more closely with procedures and controls of a spacecraft.

Acknowledgments

We wish to acknowledge the help provided by Professor Dr. Claudio Kirner for allowing initial tests in the Smart Board placed into the Educational Resources and Augmented Reality Laboratory at the Federal University of Itajubá-MG, Brazil.

We also would like to thank at INPE-DSS, Simone Cunha Léo and Denise Rotondi, simulator developers, which have constructively added suggestions during this research work.

REFERENCES

[1] INPE, National Institute for Space Research - "Instituto Nacional de Pesquisas Espaciais," 2012, Available: http://www.inpe.br/.

[2] European Space Agency Members, "ECSS", 2010, Available: http://www.ecss.nl/.

[3] T. C. Sorensen, E. J. Pilger, M. S. Wood and M.A. Nunes, "A University-developed Comprehensive Open-architecture Space Mission Operation System (COSMOS) to Operate Multiple Space Vehicles", SpaceOps 2012, Stockholm, 2012. Avaiable: http://www.spaceops2012.org/proceedings/documents/id1296468-Paper-001.pdf.

[4] S. K. Card, A. Newell and T. P. Moran, The Psychology of Human-Computer Interaction, Hillsdale, NJ: L. Erlbaum Associates Inc, 1983.

[5] A. M. Ambrosio, P. E. Cardoso, V. Orlando and J. B. Neto, "Brazilian Satellite Simulators: Previous Solutions Trade-off and New Perspectives for the CBERS Program", Proceedings of the 8th Conference on Space Operations (SpaceOps2006), 2006, Rome, AIAA.

[6] J. Tominaga, C. S. Cerqueira, J. Kono and A. M. Ambrosio, "Specifying Satellite Behavior for an Operational Simulator", Workshop on Simulation and EGSE for Space Programmes, 2012, ESTEC/Noordwijk Holland.

[7] J. Tominaga, "Satellite Simulator to Verification of Flight Plan Operation" translated of "Simulador de satélites para verificação de planos de operação em voo", Dissertation to a Master Degree at INPE, 2010, São José dos Campos, Available: http://urlib.net/8JMKD3MGP7W/37HL3J8.

[8] G. C. Burdea and P. Coiffet, "Virtual Reality Technology", New York: John Wiley & Sons, Inc., 2003.

[9] J. Wilson, "NASA", NASA, 2012, Available: http://www.nasa.gov/.

[10] Mars Pathfinder - Welcome to Mars", NASA, 1997, Available: http://mars.jpl.nasa.gov/MPF/ops/sol20-21.html.

[11] European Space Agency, "ESA", ESA, 2012, Available: http://www.esa.int/esaCP/index.html.

[12] M. Molineaux, M. Klenk and D. W. Aha, "Goal-Driven Autonomy in a Navy Strategy Simulation", Proceddings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, 2010, IDEA.

[13] E. Kavakli and P. Loucopoulos, "Goal Modelling in Requirements Engineering: Analysis and Critique of Current Methods", Information Modeling Methods and Methodologies, pp 102-124

[14] Analythical Graphics, Inc, "STK", AGI, 2012, Available: http://www.agi.com/products/.

[15] ESA-ESTEC, "Space engineering - System modeling and simulation", 2010, Noordwijk, ECSS.

[16] B. Irving, A. McSorley, M. Paton and G. Bonin, "Virtual prototyping of human Mars missions with the Orbiter space flight simulator", Mars Society Conference, 2006, Washington. Avaiable: http://download.orbit.m6.net/news/Virt_Proto_Mars_Orbiter_Irving_et_al.pdf.

[17] Celestia Development Team, "Celestia", 2010, Available: http://www.shatters.net/celestia/.

[18] A. S. Tanenbaum and M. v. Steen, "Distributed Systems: Principles and Paradigms", Pearson Education, 1995.

[19] Y. Chen, "Modeling and Simulation for and in Service-Orientated Computing Paradigm", SIMULATION: Transactions of The Society for Modeling and Simulation International, January 2007, v. 83, n. 1, pp. 3-6, SCS.

[20] J. J. M. Moreira, "WSQL : uma arquitectura de software baseada em web services", 2012. Available: http://hdl.handle.net/10216/11646.

[21] T. D. Moro, C. F. Dorneles and M. T. Rebonatto, "Web services WS-* versus web services REST", 2009, Available: http://www.upf.br/computacao/images/stories/TCs/arquivos_2009 2/Tharcis_Dal_Moro.pdf.

[22] Z. Lieberman, T. Watson and A. Castro, "oF", oF Community, 2012, Available: http://www.openframeworks.cc/.

[23] Applied Informatics Software Engineering GmbH, "POCO C++ Libraries", Applied Informatics Software Engineering GmbH, 2012. Available: http://pocoproject.org/.

[24] C. Kirner, C. S. Cerqueira and T. G. Kirner. "Using Augmented Reality Cognitive Artifacts in Education and Virtual Rehabilitation", Virtual Reality in Psychological, Medical and Pedagogical Applications, Christiane Eichenberg (Ed.), ISBN: 978-953-51-0732-3, InTech, Available from: http://www.intechopen.com/books/virtual-reality-in-psychological-medical-and-pedagogical-applications/using-augmented-reality-cognitive-artifacts-in-education-and-virtual-rehabilitation

[25] SMART Technologies, "SMART," SMART Technologies, 2012, Available: http://smarttech.com/.

[26] F. Antonio, "Keplerian Elements Tutorial", AMSAT 2001, Available: http://www.amsat.org/amsat/keps/kepmodel.html.

[27] N. T. O. Reis, "Fundamentos de Mecânica Orbital 1", Available: http://educacaoespacial.files.wordpress.com/2011/08/mecc3a2nicaaa3.pdf

[28] Kaplan, M.H. (1976). Modern Spacecraft Dynamics and Controls. Wiley, New York. ISBN 978-0-471-45703-9