THE JASMINE SIMULATOR

Yoshiyuki Yamada, the JASMINE Working Group

Department of Physics, Kyoto University, Kyoto 606-8502, JAPAN

ABSTRACT

We explain the framework of simulation tools in the JASMINE project. The JASMINE project stands at the stage where its basic design will be determined in a few years. Then it is very important to simulate the data stream generated by astrometric fields in JASMINE in order to support investigations of error budgets, sampling strategy, data compression, data analysis, scientific performances, etc. Of course, component simulations are needed, but total simulations which include all components from observation target to satellite system are also very important. We find that new software technologies, such as Object Oriented (OO) methodologies are ideal tools for the simulation system of JASMINE (the JASMINE simulator).

Key words: JASMINE; Simulator.

1. INTRODUCTION

The JASMINE project stands at the stage where its basic design will be determined in a few years. Then it is very important to simulate the data stream generated by astrometric fields at JASMINE in order to support investigations of error budgets, sampling strategy, scientific performances, etc. The simulation system should include all components in JASMINE as 'objects' in order to resolve all issues which can be expected beforehand and make it easy to cope with some unexpected problems which might occur during the JASMINE mission. Components include not only models of concrete materials such as scientific instruments and the satellite bus system, but also abstract ones such as observation methods, orbits, data transfer, algorithms of data analysis etc. Furthermore many researchers in various fields including engineering of the bus system should participate in developing the simulator. In that case the simulator includes many objects and furthermore flexibility and interaction is needed to cope with the objects. On the other hand, calibrations must be protected from unintentional modification for each model of the objects. Then we conclude that the Object Oriented methodologies are ideal for the demands described above in the simulator. High maintainability and reusability of the object oriented approach is very useful in making the simulator system. We are now constructing the JASMINE simulator using techniques developed in the field of information science.

In Section 2, the requirements of the JASMINE simulator are discussed. Section 3 contains some examples of the JASMINE system investigations with the simulator software. Section 4 is devoted to the future schedule on software development.

In this paper we will make reference to terms with specific meaning within an object-oriented programming context. These are:

class encapsulates common behavior of a group of objects;

attributes data member of a class;

methods functions which may be performed on instances of a class;

object an instance of a class;

abstract class a class from which no instances may be created;

inheritance a way to form new classes or objects using predefined objects or classes where new ones simply take over old one's implementations and characteristics.

2. SPECIFICATION REQUIREMENT OF THE FRAMEWORK

Implementation of the simulator is carried out in two steps. In the first step, we construct or choose the framework of the software. The second step is building software which is JASMINE specific under the framework built in the first step. In this section, we discuss the framework of the JASMINE simulator.

The JASMINE simulator will be mainly used for assessing the scientific goals of the mission, instrument design investigation, data treatment preparation, and algorithm study such as stellar image detection or evaluating astrometric parameters from raw data. The need for a clear, modular and easily communicable structure for the simulator led us to choose an object-oriented language for its implementation.

In the simulator, each component of JASMINE is treated as an ‘object’ which has specifications as attributes. For example, an object which represents a CCD detector has attributes such as pixel size, number of pixels, quantum efficiency, flatness, charge transfer rate, etc. Such attributes affect the final scientific outputs.

Within the context of system investigations, some attributes of some components may depend on other attributes of other components. We can draw diagrams which represent dependencies of each component (object) by analyzing the above dependencies. It is called a ‘Data Flow Diagram’. The framework is expected to provide the function of creating and editing such ‘data flow’ diagrams.

The ‘data flow’ during JASMINE observations begins with an astronomical object, is related to an optical system, detectors, and data-processing units, and ends as a communication system. One simple implementation is specifying the relation of abstract classes which represent such components beforehand. Then, the framework can deal with detailed investigations easily by making concrete classes which are inherited from the abstract classes. The following calculation is performed in the flow of the JASMINE specification study,

$$f = \frac{N_{\rm PSF} w D}{\lambda},$$

where f, w, D, λ are the total focal length of the optical system, pixel size of the CCD, diameter of a primary mirror, and wavelength respectively, and $N_{\rm PSF}$ is a constant. An attribute of optical system f depends on an attribute of detector w. These dependencies (‘data flow’) have the opposite direction of a real ‘data flow’ in observation. One of the requirements of our simulator is that we can specify the direction of ‘data flow’ in our diagram as both in the same and counter direction as ‘data flow’ of a real observation.

From the above considerations, the framework of the JASMINE simulator may have functions for managing DAG (directed acyclic graph). To summarize the requirements:

1. computational modules are extendable,
2. data types are extendable,
3. open source (if using existing software),
4. some simple graphical tools are provided,
5. platform independent,
6. availability of treating distributed computation via network.

Requirements (1)-(3) are critical and (4)-(6) are optional.

A schematic UML diagram of our prototype simulator is shown in Figure 1. In our implementation of a prototype simulator, each component, such as optical systems or detectors, is inherited from an abstract class ‘model’. A

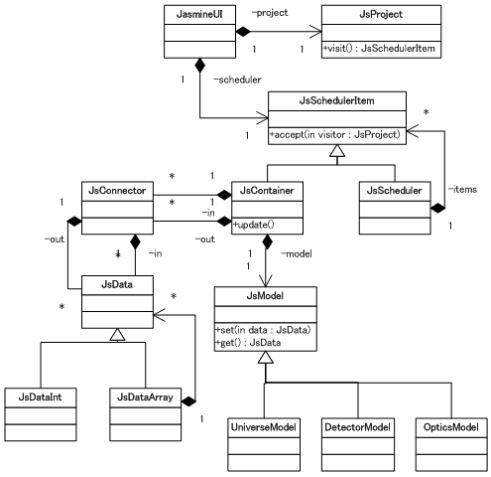

Figure 1. UML diagram of the framework of prototype simulator code. Names of classes JsModel, JsContainer, JsConnector, and JsData correspond to ‘model’, ‘container’, ‘connector’, and ‘data’ in the main text.

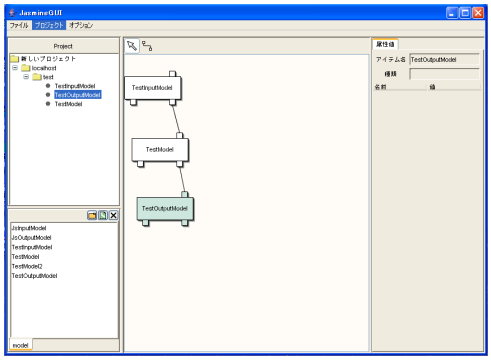

Figure 2. Start-up view of the JASMINE Simulator prototype.

class ‘container’ which manages dependency graph has the class ‘model’. The structure ensures not also extendability of modules but the arbitrariness of the direction of dependency. Each directed arc is represented as a class ‘connector’, which encapsulates flowing ‘data’ between two ‘models’. The abstraction ‘data’ ensures extendability of data. The model can be easily extendable by making subclasses of ‘model’.

3. CONTENTS OF OUR SIMULATOR

In this section, we show implemented results and implementation plans of JASMINE specific components. We have already performed several items:

- observational method and accuracy,
- data reduction scheme,
- stellar image detection,
- data compression scheme,

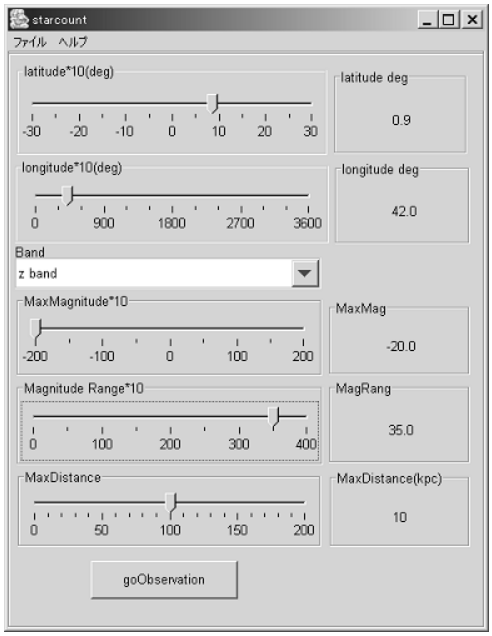

Figure 3. StarCount Simulator.

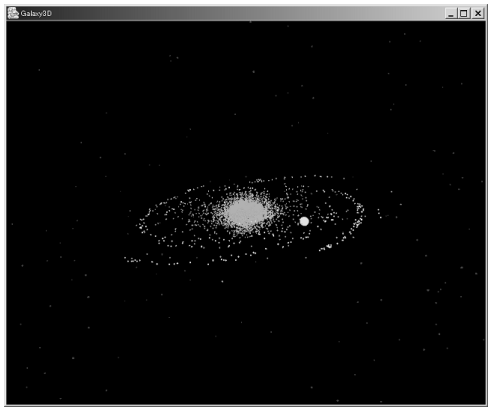

Figure 4. 3D Galaxy view.

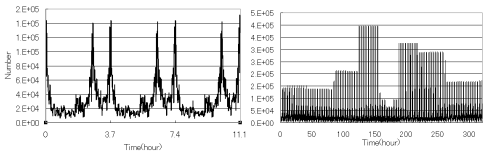

Figure 5. Time variation of Star Number

- star number counts,
- optical system performance,
- satellite attitude and accuracy,
- orbits,
- etc.

In this section, we discuss two topics: the Galaxy model and stellar count numbers, and simulations of the CCD and focal plane.

3.1. Galaxy Model and Stellar Number Counts

The main targets of the JASMINE project are stars in the Galactic disc and bulge. There is no complete observation which shows number density of stars brighter than specified magnitude in the z-band and contained in the Galactic disc and bulge. Thus, it is important to forecast numbers of stars, data rate etc using the model Galaxy which is well confirmed from observations. We use Wainscoat et al. (1992) and Cohen (1994) for the model of stellar number distribution in the Galaxy. This model well reproduces 2MASS results. We also take motion of stars from the Evans model, and the distribution of dust by COBE DIRBE data into account.

We implemented Star Count Simulator (SCS) first (see Figure 3). This is the tool for calculating the number of stars with arbitrary direction, arbitrary distance, and arbitrary spectrum types. This enable us to evaluate the number of stars observed at every moment. We also implemented a three-dimensional viewer of the Galaxy model (Figure 4).

Figure 5 shows the time variation of the number of stars observed by JASMINE. We cannot estimate the amount of processing by average stellar density because the variability of numbers is very large. Such information is very important for designing data processing unit and data recorders.

3.2. Modelling of CCD and Focal Plane Simulation

We have also started on implementing a CCD model. The CCD can be modelled as the module which has many accumulator (pixel), data transfer mechanisms, and reading mechanisms. Efficiency of accumulation (quantum efficiency) is about 0.9 and depends on the wavelength. Data (charge) transfer efficiency is extremely close to unity but is not unity. Reading data has also some errors. These effects are implemented as a probability process on the CCD model. Satellite environments such as cosmic rays may affect such probabilities.

Using our Galaxy model and satellite attitude model, a list of stars contained in two fields of view are generated. According to the attributes of each star such as distance, spectral types, etc, photons are generated using a probability process. The astrometric optical system has a very long focal length. The point spread functions are broad compared with the pixel size. To generate images on the focal plane also requires the use of a probability process. From the above several probabilities processes, ‘times’ and ‘locations in detector plane’ pairs belonging to each photon are generated. Each ‘photon’ may be the trigger of the CCD model. By each trigger, the CCD model accumulates ‘charge’ of the appropriate ‘pixel’. Continuing a pseudo exposure during an appropriate interval, we can get a pseudo image of a JASMINE observation.

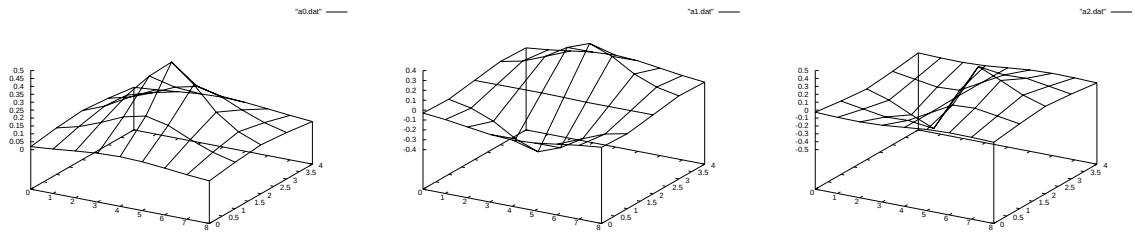

Figure 6. The first three principal modes of stellar image.

3.3. Data Compression

Table 1. Original data size is 720 bit. Compression rate are approximately 60%.

N†	residue	bit (pv)	bit (k)	total
0	338.527	0	4	342.527
1	247.013	12.247	4	263.260
2	222.109	22.737	4	248.846
3	207.794	31.814	4	243.608
4	203.974	39.257	4	247.231
5	202.063	45.703	4	251.766
6	200.866	51.495	4	256.361
7	199.738	57.001	4	260.739
8	198.485	62.497	4	264.982

†: Number of principal value

Combination of Karuhnen-Loeve transformation and Golomb-Rice code is suitable for compression of stellar image data. We construct bases of Karuhnen-Loeve transformation from 10 000 ideal pseudo data. The first three principal components correspond to magnitude and two coordinates of the centroid. In this method, we achieve 60% compression for each 720 bit data.

4. CONCLUSION

We have already performed several investigations on the JASMINE system design using Fortran, C++ and Java codes.

ACKNOWLEDGMENTS

We would like to acknowledge Y. Kawakatsu, A. Noda, A. Tsuiki, M. Utashima, A. Ogawa, N. Sakou, H. Ueda, H. Okumura for their collaboration in the investigations on the JASMINE spacecraft system. Furthermore we would like to thank Miyashita Hisashi at IBM Tokyo Research for useful advice on general code construction technique. This work has been supported in part by the Grant-in-Aid for the Scientific Research Funds(15340066) and Toray Science Foundation.

REFERENCES

Wainscoat, R.J., Cohen, M., Volk, K., Walker, H.J. & Schwartz, D.E.,1992, Astrophys.J.Suppl. 83, 111

Cohen, M.,1994, Astrophys.J. 107, 582