特集2

# キャッツの海外展開の現状と今後

## ■はじめに

キャッツでは以前より、国内だけでなく海外にも視野を広げてきた。今回の資本提携を受け、キャッツでは更なる海外展開を進める活動を実施している。その活動の一つとして、海外企業にキャッツのツール製品を評価してもらっている。

ヨーロッパ、北米からは、デンマーク、スウェーデン、ギリシャ、フランス、アメリカ、カナダなどの各国の組込み製品に携わっている企業や研究所に興味を持ってもらっており、その中でもすでに評価を終了している企業からは、高評価をもらっている。

ギリシャ (社名:ISD)の企業には、キャッツのXModelinkツールの評価を実施してもらった。評価指標のベースは、ISO9126でそれぞれ、直観的に理解できる操作環境であることを示す、学習容易性、目的としている作業内容が実現可能であることを示す機能性、目的としている作業内容が短時間でできることを示す時間効率性、エラーの内容が理解できることを示すエラー理解性、全体の使用に関する使用性、ツールの反応などの有効性を示す性能、データ損失しない安心感を示す信頼性、見やすい・親しみやすいなどを示す好感度の8項目とした。(図1参照)　また、この項目以外にも、具体的なよかった点、悪かった点、改善するべきポイントなどの意見をもらった。今回、開発プロセスの設計書作成に関しては、非常に高評価な意見をもらうことができた。しかし、多少の改善を要するコメントも見られたので、ツール自体の品質向上、有効性向上に役だてたいと思う。詳細に関しては、キャッツWEBページからダウンロード可能なので、是非参考にしてもらいたい。

また、アジアからは中国、韓国の企業や大学、研究所に興味を持ってもらい、ツールの評価を実施している。今年度は、すでに中国の北京や大連で開かれた組込みの展示会に出展し、会場でも大きな反響を得ることができた。今後は、ヨーロッパや北米のみならず、アジアの各国へ更なる活動を実施していく。

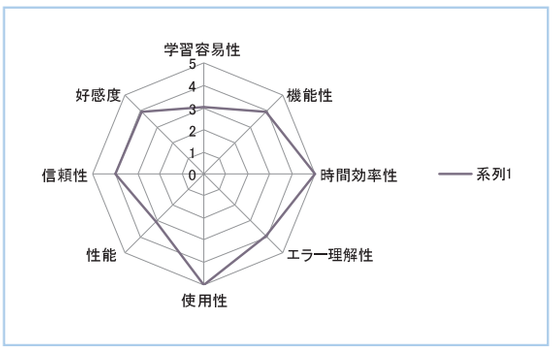
## 1. ISD社(ギリシャ)の弊社ツール「XModelink」の評価表

以下は、実際に上記指標にそってツールを評価してもらった結果である。

ツール評価質問票　（Xmodelink）

| | | 評価 | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 1～5 |
| 番号 | 項目 | 低い | やや低い | 普通 | やや高い | 高い | 結果 |
| 1 | 直感的に理解できる操作環境である（学習容易性）Ease of learning | | | 3 | | | 3 |
| 2 | 目的としている作業内容が実現可能である（機能性）Functionality | | | | 4 | | 4 |
| 3 | 目的としている作業内容が短時間でできる　（時間効率性）Time effectivene | | | | | 5 | 5 |
| 4 | エラーの理解ができる　（エラー理解性）Comprehension of error | | | | 4 | | 4 |
| 5 | 使用性　Usability | | | | | 5 | 5 |
| 6 | 反応がよい（性能）Performance | | | 3 | | | 3 |
| 7 | データ喪失しない安心感がある　（信頼性）Reliability | | | | 4 | | 4 |
| 8 | 見易い・親しみ易い　好感度（好感度）Favorability | | | | 4 | | 4 |
| 9 | | | | | | | |
| 10 | | | | | | | |

| ユーザ要望 |
|---|
| よかった点、改善を要する点について記入してください。 |
| よかった点 |
| Useful towards the identification of design issues in the development process. |
| 開発プロセスの設計書作成に関しては、使える。 |
| |
| 改善を要する点 |
| Little documentation in English. |
| 英語のドキュメントが少ししかない。 |




［図1］

### 2. 適用評価レポート

海外のツール評価では、先方より任意の形式で、実際の開発に沿ったレポートをもらっている。
今回の内容は以下とおりである。

## Abstract

This brief report provides feedback of an evaluation performed by ISD to the Xmodelink SystemC Tool. It also itemizes possible extensions that also aim towards advanced visualization.

# 1 XModelink Evaluation Process

## 1.1 Purpose and Main Components of the Tool

The purpose of the XModelink Tool is to provide some assistance to the SystemC developer on identifying the functionality of his SystemC design throughout the development process while also detecting bugs early in the lifecycle. The tool is composed of two sub components:

- the thread viewer illustrating SystemC threads during simulation

- the vcd viewer illustrating signal traces that are stored in vcd waveforms.

## 1.2 Test Environment

Before evaluating the XModelink Debugging Tool, Microsoft Visual Studio 2005 and systemc-2.1 were installed on a Windows Workstation.

## 1.3 SystemC Benchmarks used

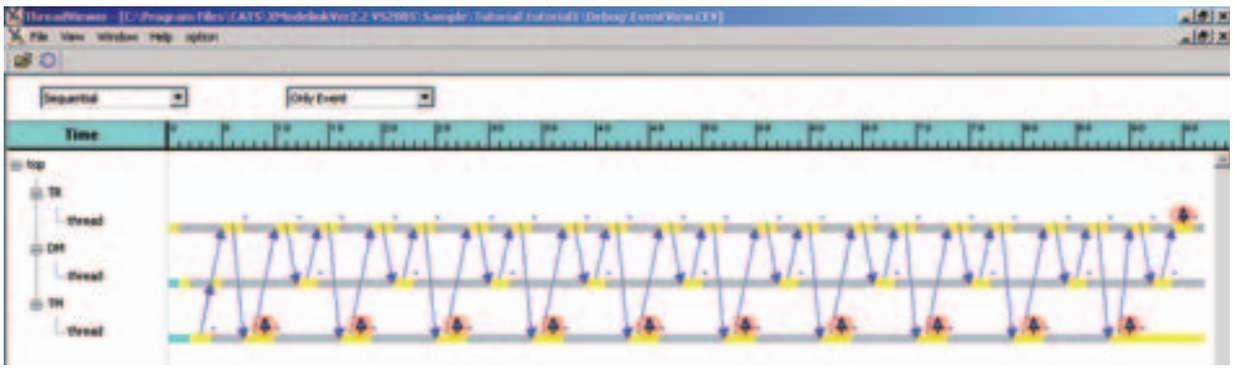
The first step towards examining the usefulness of the Tool involved running Tutorials 1 and 2 provided with CATS' XModelink Installation, followed by using the tool on other ISD SystemC designs.

In Tutorial 2 of installation there are three blocks a test module, data module and transceiver module. The test module feeds a transceiver with data values incremented by 2 on each edge of the clock cycle. The test data is sent from the transceiver to the data module, while the latter (data module) saves this data on a local variable. Signals entering and exiting the transceiver module are traced on a waveform.

The vcd viewer allows the user to observe traces that were inserted in the SystemC design.


*[Figure 1.1] XModelink vcd viewer – Tutorial 2*

In Tutorial1, there are once again three blocks the test module, data module and transceiver module. This time communication between modules is performed using threads. Testmodule sends test data to transceiver module and initiates a notification start. This notification goes both to to transceiver and data modules. With the aid of additional events, the Transceiver forwards input data to the data module while the latter saves it to a local variable.
The XModelink viewer in this case allows the SystemC developer to view the execution of threads in an easy to understand graphical manner.




*[Figure 1.2] XModelink thread viewer – Tutorial 1*

On the second phase of evaluation the product was used in the debugging of a complex NoC architecture. The design was compiled and ran under Visual Studio and design execution was monitored using sc_traces of signals and events. The XModelink thread viewer was found particularly useful in this case. It reveal bugs that were not obvious at first sight during development and required visualization for understanding of how problem occurs. Details on the NoC architecture however cannot be made public.

## 1-4 General Conclusions and Extensions

The XModelink tool was used to evaluate provided a set of tutorials as well as designs that reside to company's IP. The evaluation showed that XModelink allowed in depth understanding of complex SystemC execution revealing design errors early in the design process in a clear manner. Some fresh ideas for possible extensions to Xmodelink tool especially towards data introspection capabilities are listed below.

The SystemC debugger can be extended towards SystemC-AMS modeling, providing ultimately a unified configurable environment applicable to interoperable software, digital, discrete-time/event and analogue/RF macroarchitecture models. This environment should provide enriched testing of models based on non-intrusive monitoring and powerful debug/analysis facilities providing system-level exchange of information between the models and the designer at different levels of abstractions.

For large multicore SoC models, an integrated debugging environment that facilitates system designers in simulating, debugging, and visualizing SystemC models should integrate efficient system-level debugging with advanced design space exploration and visualization features.

More specifically, efficient system-level debugging must combine user model error detection, diagnosis (understanding, localization) and correction with SystemC-aware debugging based on kernel visibility, e.g. using non-intrusive abstractions of internal signals, ports, events, processes and modules. In addition, advanced model design space exploration and system model and platform visualization features must provide simulation displays at different abstraction levels, history, visibility of interactions, multiple hierarchical views, status/event reports, high-level model and system power-performance metrics and animation features with monitor/trace controls, model-specific command/control messaging, and save/restore/traceback functionality.

## *WEBサイトにて海外ユーザの評価レポート公開予定*

**・この他にも様々なツールのレポートを続々配信！**

http://www.zipc.com/instance/