行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※ 具可延展性嵌入式核心程式及其資源排程技術 ※
※ 之研究與設計(II) ※
※ **Researches and Design of an Extensible,** ※
※ **Embedded Kernel and Its Resource Scheduling** ※
※ **Techniques (II)** ※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：☑ 個別型計畫　　☐ 整合型計畫

計畫編號：NSC　90-2213-E-009-019

執行期間：90 年 8 月 1 日至 91 年 7 月 31 日

計畫主持人：　張瑞川　　國立交通大學資訊科學學系

計畫參與人員：　李岳峰　　國立交通大學資訊科學學系

康容致　　國立交通大學資訊科學學系

汪國駒　　國立交通大學資訊科學學系

本成果報告包括以下應繳交之附件：

☐赴國外出差或研習心得報告一份

☐赴大陸地區出差或研習心得報告一份

☐出席國際學術會議心得報告及發表之論文各一份

☐國際合作研究計畫國外研究報告書一份

執行單位：國立交通大學資訊科學學系

中　華　民　國九十一年五月十七日

行政院國家科學委員會專題研究計畫成果報告
具可延展性嵌入式核心程式及其資源排程技術之研究與設計(II)
Researches and Design of an Extensible, Embedded Kernel and Its Resource Scheduling Techniques (II)

計畫編號：NSC 90-2213-E-009-019
執行期限：90 年 8 月 1 日至 91 年 7 月 31 日
計畫主持人：　張瑞川　國立交通大學資訊科學學系
計畫參與人員：　李岳峰、康容致、汪國駒
國立交通大學資訊科學學系

一、中文摘要

近年來由於消費性電子產品市場的崛起，帶動國內外對嵌入式系統（Embedded System）及其相關環境的研究。然而，目前嵌入式系統仍存在著一個嚴重的問題：缺乏延展性。

80 年代被提出的微核心架構(micro-kernel) 雖然使核心程式較容易擴充及調整，但它有效能上的問題。而 90 年代被提出的可延展式核心架構（extensible kernel architecture）試圖改善微核心架構的效能問題並提升系統的可延展性。然而，它主要的議題皆在探討如何讓應用程式調整系統服務之餘，系統仍能保持正常的運作與效能。有關嵌入式系統資源有限(resource-limited)的問題並不在其探討的範圍內。

本計畫預計以三年時間探討新一代的具可延展性的嵌入式系統架構。藉由本計畫，我們希望能擷取嵌入式系統，可延展式核心架構，以及元件化作業系統的優點，開發新一代更適合於嵌入式設備的作業系統架構及資源排程技術。

在計畫的第一年中，我們研究並製作了一個具可延展性的嵌入式系統架構。它可以讓應用程式選擇需要的系統功能及策略元件。在今年度（第二年）的計畫中，我們實作了核心程式元件的動態下載機制。包含嵌入式核心程式內動態連結程式（Dynamic Linker）的實作、元件入口站的設計、及嵌入式設備與入口站間通訊協定的設計。在下一年度我們將探討動態下載技術的安全性問題，及研究嵌入式系統上資源排程的技術。

關鍵詞：作業系統、嵌入式系統、可延展性系統、軟體元件

Abstract

In recent years, raise of market for comsumption-electronic products has pushed the researches on embedded systems and their associated environments. However, there is a drawback for current embedded systems: lack of extensibility.

Micro-kernel architecture was proposed in Eighties to make kernels more extensible. But it has performance problems. Extensible kernel architectures, which are proposed in Nineties, aimed at improving the performance of micro-kernels as well as making systems more extensible. However, the major issue is to allow applications extend the services provided by the kernels while the system integrity and performance are still maintained. The problem of resource limitation in embedded systems is not addressed.

In this project, we plan to spend 3

years researching on next generation, extensible embedded system architecture. We intend to catch the strength of embedded systems, extensible kernels and component-based systems, and to develop a system architecture and an integrated resource scheduling framework that are suitable for next generation embedded devices.

In the first year of the project, we implemented an extensible embedded system architecture, which allows applications to select system functionality and components according to their needs. In this year (i.e. the second year), we designed and implemented the dynamic downloading framework for system components. The framework includes the dynamic linker, the component portal, the communication protocol between the embedded devices and the portal. In the next year, we will put focus on the security problems of the framework and the integrated resources scheduling techniques on the embedded systems.

Keywords：Operating System, Embedded System, Extensible System, Software Component

二、計畫緣由與目的

近年來作業系統在應用市場方面有二項重要的發展。第一項是 Linux 及 Open Source 的風潮，這對於作業系統的研究及開發有著莫大的助益。第二項發展則是消費性電子產品市場的崛起。隨著微電子和通訊網路技術的進步，諸如 smart phone, Set-Top Box, PDA…等嵌入式設備如雨後春筍般出現，進而帶動國內外對嵌入式系統（Embedded System）及其相關環境的研究。然而，目前嵌入式系統仍存在著一個嚴重的缺失：缺乏延展性。目前嵌入式系統所提供的系統服務（system services）是固定的，應用程式無法根據其需求動態地調整這些系統服務。例如：應用程式無法更換緩衝快取置換策略 (buffer cache replacement policy), 磁碟排程演算法(disk scheduling algorithms)。此外，嵌入式系統也缺乏一套完整的資源排程機制。傳統的嵌入式系統雖然很多都宣稱有即時(real-time)的特性，但大多只是提供一個 CPU 即時排程（real-time CPU scheduler）而已。然而，要支援真實世界中的即時及多媒體應用，光靠這樣是不夠的。除了 CPU 之外，即時及多媒體應用所牽涉到的系統資源還包含了 I/O，記憶體，中斷等。所以，如何有效地作多種資源的管理，並整合即時排程的控制，就成為一門重要的研究課題。

90 年代被提出的的可延展式核心架構（extensible kernel Architecture）試圖提升系統的可延展性。它允許應用程式根據其需求動態調整系統的服務。然而，它主要的議題皆在探討如何讓應用程式調整系統服務之餘，系統仍能保持正常的運作與效能。有關嵌入式系統資源有限(resource-limited)的問題並不在其探討的範圍內。

作業系統的架構曾經出現過幾次變革。例如：80 年代的微核心作業系統（Micro Kernel Operating System），90 年代的可延展式核心架構（extensible kernel Architecture），物件導向（Object-oriented）及元件化（component-based）作業系統等。藉由本計畫，我們希望能擷取嵌入式系統，可延展式核心架構，以及元件化作業系統的優點，探討新一代更適合於嵌入式設備上的作業系統架構。

本計畫預計以三年時間探討新一代的具可延展性的嵌入式系統架構。我們將專注於彈性化及可下載的系統服務，我們希望應用程式不再侷限於固定的系統服務而可以選擇自己所需要的。舉例而言，應用程式可選擇不要檔案系統，且 CPU 排程用 EDF。除此之外，應用程式所選擇的系統服務及策略模組皆於應用程式

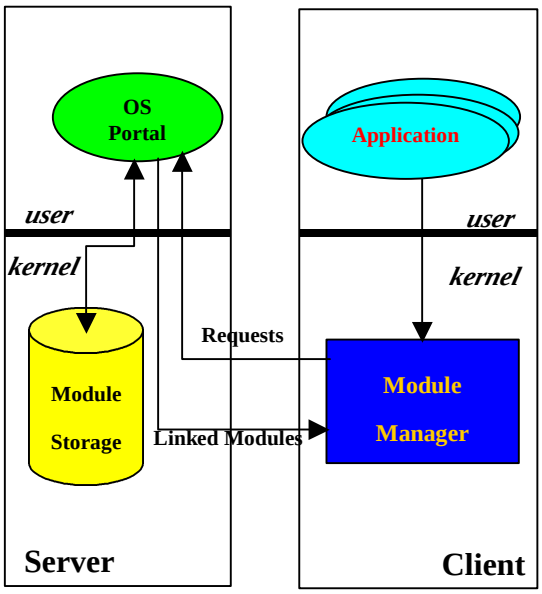
執行時動態從伺服器端下載並連結。我們以交通大學資訊科學系系統實驗室這二年在國科會支持下開發的一個元件化的核心程式作為實驗平台。基於我們之前對元件化核心程式的研究成果，我們會將每一系統服務切割成一獨立的元件以利動態下載、替換及刪除。此架構可支援各式各樣的應用程式，因為伺服器端有大量的資源空間，可存放各種可能的系統服務。同時，此架構不會對嵌入式設備造成負擔，因為只有真正需要的系統服務才會下載到嵌入式設備上。除此之外，我們也將基於此平台來對整合性的資源排程技術作深入的分析及研究。

三、結果與討論

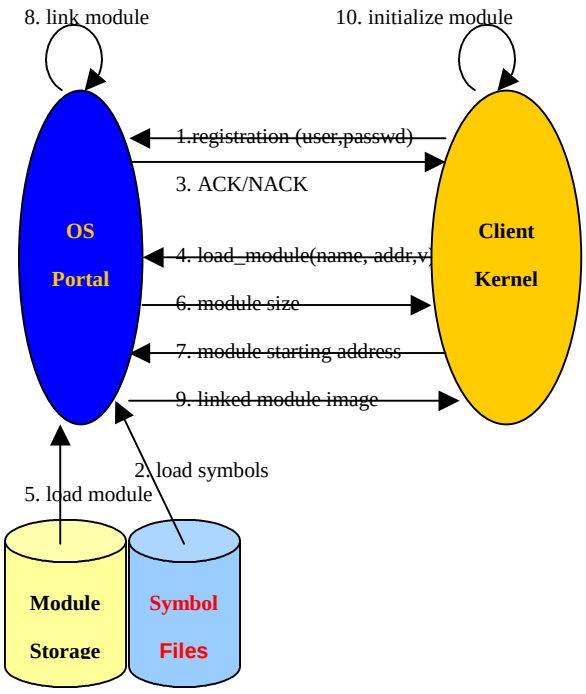
在第一年的計畫中，我們已經設計好整體的系統架構。在這一年度(第二年)，我們開始實作動態下載的機制。其中主要包含：

1. 嵌入式核心程式內動態連結程式（Dynamic Linker）的實作
2. 元件入口站的設計
3. 嵌入式設備與入口站間通訊協定的設計

系統概觀架構如圖一。在動態連結程式（Dynamic Linker）的實作方面，我們分析 Linux Kernel Module 及 user level 動態連結程式的原始碼。並以此為基礎，設計我們的動態連結程式。最後將我們的動態連結程式拆成兩個部分：一個是 client 端的 Module Manager，一個是 server 端的 OS Portal。我們將大部分負載重及耗費系統資源的工作都放在 server 端作。如此一來，client 端既具可擴充性又可大大節省系統資源。至於元件入口站，其功能主要由 OS Portal 負責。經實驗證明，我們的元件入口站可以支援每秒上千個要求，這對一般企業而言是非常足夠的了。而嵌入式設備與入口站間通訊我們目前是用 TCP/IP。因為 TCP/IP 是可信賴的傳輸協定，讓我們不用去處理資料在網路中傳輸可能會遺失的問題。這樣可以簡化我們系統在初期設計的複雜度。其協定如圖二。我們系統的優點在於：我們的動態下載機制對系統資源要求遠比其他傳統方式低(甚至只有 1%)，所以非常適用於嵌入式系統內。



圖一、核心程式動態下載系統概觀架構



圖二、嵌入式設備與入口站間之通訊

在下一年度我們將探討動態下載技術的安全性問題。這些問題包含：元件入口站如何得知與其溝通的嵌入式設備是合法使用者？嵌入式設備如何相信下載的元件不會對嵌入式設備造成破壞？等等問題。這些問題或多或少可用加密或數位簽章的方式來解決。但其所花的代價也許會太高，因為這些方式的運算複雜度都很高，貿然加入系統終將使系統的效能大幅滑落。所以我們必須尋求出一個較合理的解決方式。此外，我們也將以我們的動態可延展性核心為實驗平台，研究嵌入式系統上資源排程的技術。

四、參考文獻

[1] A. C. Veitch and N. C. Hutchinson, "Kea - A Dynamically Extensible and Configurable Operating System Kernel", Proc. 3rd Conference on Configurable and Distrib-uted Systems (ICCDS'96), Annapolis, Mariland (1996).

[2] A.Gotz, P.Makijarvi, B.Regad, M.Perez, P.Mangiagalli, "Embedding Linux to Control Accelerators and Experiments", Linux Journal, issue 66, October 1999.

[3] Brian Bershad, S. Savage, P. Pardyak, E. G. Sirer, M.Fiuczynski, D. Becker, S. Eggers, C. Chambers., "Extensibility, safety and performance in the Spin operating system", In 15th ACM Symposium on Operating System Principles, pages 267-284, Copper Mountain Resort, Colorado, December 1995.

[4] Bryan Ford, Godmar Back, Greg Benson, Jay Lepreau, Albert Lin, Olin Shivers, "The Flux OSKit: A Substrate for Kernel and Language Research", In Proceedings of the 16th ACM Symposium on Operating Systems Principles, pages 38-51. ACM SIGOPS, Saint-Malo, France, October 1997.

[5] Chris Maeda and Brian Bershad, "Protocol Service Decomposition for High-Performance Networking", In 14th ACM Symposium on Operating System Principles, pages 244-255, 1993.

[6] D. Engler, M. Frans Kaashoek, J. O'oole Jr., "Exokernel: An Operating System Architecture for Application-Level Resource Management" Proc. 15th SOSP, pp. 251-266 (1995).

[7] D. Golub, R. Dean, A. Forin, R. Rashid, "UNIX as an Application Program," Proc. 1990 Summer USENIX, pp. 87-96 (1990).

[8] D. Hildebrand. An Architectural Overview of QNX. In Proc. of fhe USENIX Workshop on Micro-kernels and Other Kernel Architectures, pages 113-126, Seattle, WA, Apr. 1992.

[9] Daniel Julin, Jonathan Chew, Mark Stevenson, Paulo Guedes, Paul Neves, Paul Roy, "Generalized Emulation Services for Mach 3.0: Overview, Experiences and Current Status", In Proceedings of the Usenix Mach Symposium, 1991.

[10] Daniel S. Decasper, "A software architecture for next gen-eration routers", PhD thesis, Swiss Federal Institute of Technology, Zurich, 1999.

[11] David Black, David Golub, Daniel Julin, Richard Rashid, Richard Draves, Randall Dean, Alessandro Forin, Joseph Barrera, Hideyuki Tokuda, Gerald Malan, David Bohman. "Microkernel Operating System Architecture and Mach", In 1st USENIX Workshop on Micro-kernels and Other Kernel Architectures, pages 11-30, Seattle, April 1992.

[12] ELKS: The Embeddable Linux Kernel Subset, Available as http://www.elks.ecs.soton.ac.uk/

[13] Eran Gabber, Christopher Small, John Bruno, Jose Brustoloni and Avi Silberschatz, "The Pebble Component-Based Operating System", Proceedings of the 1999 USENIX Annual Technical Conference, Monterey, CA, USA, June 6-11, 1999.

[14] Integrated Systems, Inc.,"pSOSystem Technical Overview", http://www.isi.com/products/psosystem/tech_over.pdf

[15] J. Liedtke, "On Micro-Kernel Construction," Proc. 15th SOSP, pp. 237-250 (1995).

[16] M. Rozier, V. Abrossimov, F. Armand, I. Boule, M. Gien, M. Guillemont, F. Herrmann, C. Kaiser, S. Langlois, P. Leonard, W. Neuhauser. "Chorus Distributed Operating System." Computing Systems 1(4), pp. 305-370 (1988).

[17] N. C. Hutchinson and L. L. Peterson. "The x-kernel: an architecture for implementing network protocols", IEEE Trans. Software Engineering, 17(1):64-76, January 1991.

[18] R. Campbell, N. Islam, P. Madany, and D. Raila,"Designing and Implementing Choices: An Object-Oriented System in C++", Communlcations of the ACM, Sept. 1993.

[19] R. Pike, D. Presotto, K. Thompson, H. Trickey, "Plan 9 from Bell Labs," Proc. Summer 1990 UKUUG Conf., pp. 1-9 (1990).

[20] Richard Draves, Scott Cutshall. Unifying the User and Kernel Environments. Microsoft Research Technical Report MSR-TR-97-10, 16 pages, March 1997. Available from ftp://ftp.research.microsoft.com/pub/tr/tr-97-10.ps.

[21] Robert Morris, Eddie Kohler, John Jannotti, and M. Frans Kaashoek, "The Click modular router", Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP 99), Kiawah Island, December 1999.

[22] S. Dorward, R. Pike, D. Presotto, D. Ritchie, H. Trickey, P. Winterbottom, "Inferno," Proc. IEEE Compcon 97,pp. 241-244 (1997).

[23] S. Goel and D. Duchamp, "Linux DeviceDriver Emulation in Mach", In Proc. of the Annual USENIX 1996 Technical Conference, pages 65-73, San Diego, CA, Jan. 1996.