The Internals of Distributed TyCO

Luís Lopes
Departamento de Ciência de Computadores
Faculdade de Ciências, Universidade do Porto

joint work with

Departamento de Informática
Faculdade de Ciências, Universidade de Lisboa

Talk Outline

- The DiTyCO Calculus
- The Programming Language
- The Computational Model
- Current and Future Work

The TyCO Calculus • Syntax

x			channels
X			procedures
l			method labels
P	$::=$	$\mathbf{0}$	terminated process
	$\mid$	$P \mid P$	concurrent composition
	$\mid$	**new** $x\ P$	new local variable
	$\mid$	$x!l[\tilde{v}]$	asynchronous message
	$\mid$	$x?\{l_1(\tilde{x}_1) = P_1, \ldots, l_n(\tilde{x}_n) = P_n\}$	object
	$\mid$	**def** $X_1(\tilde{x}_1) = P_1\ \ldots\ X_n(\tilde{x}_n) = P_n$ **in** P	definition
	$\mid$	$X[\tilde{v}]$	instantiation

The DiTyCO Calculus • Syntax

s			sites
P	$::=$	$\mathbf{0}$	terminated process
	$\mid$	$P \mid P$	concurrent composition
	$\mid$	**new** $x\ P$	new local variable
	$\mid$	$x!l[\tilde{v}]$	asynchronous message
	$\mid$	$x?\{l_1(\tilde{x}_1) = P_1, \ldots, l_n(\tilde{x}_n) = P_n\}$	object
	$\mid$	**def** $X_1(\tilde{x}_1) = P_1\ \ldots\ X_n(\tilde{x}_n) = P_n$ **in** P	definition
	$\mid$	$X[\tilde{v}]$	instantiation
N	$::=$	$\mathbf{0}$	terminated network
	$\mid$	$N\ \|\|\ N$	concurrent composition
	$\mid$	**new** $x@s\ N$	new local variable
	$\mid$	**def** $D@s$ **in** N	definition
	$\mid$	$s[P]$	site running process

(cf. talks on LSDπ by Francisco Martins and António Ravara)

The Base Calculus • Operational Semantics

- reduction (always local, within sites):

Communication	$x?\{\ldots, l(\tilde{x}) = P, \ldots\} \mid x!l[\tilde{v}] \rightarrow \{\tilde{v}/\tilde{x}\}P$
Instantiation	**def** $X(\tilde{x}) = P$ **in** $X[\tilde{v}] \mid Q \rightarrow$ **def** $X(\tilde{x}) = P$ **in** $\{\tilde{v}/\tilde{x}\}P \mid Q$

- weak migration (site-to-site, triggered by lexical scope):

Message Migration	$r[x@s!l[\tilde{v}]] \rightarrow s[x!l[\tilde{v}\sigma_{rs}]]$
Object Migration	$r[x@s?M] \rightarrow s[x?M\sigma_{rs}]$
Remote Instantiation	**def** $D@s$ **in** $r[X@s[\tilde{v}]] \rightarrow$ **def** $D@s$ **in** $s[X[\tilde{v}\sigma_{rs}]]$

The Programming Language

Abstractions:

- straightforward extension of TyCO;
- sites are the programming modules;
- interface channels are explicitly located at sites;
- **export** defines the external site interface;
- **import** establishes bindings with remote resources.

Semantics:

- channels and procedures are the distributed resources;
- migration triggered by lexical scope;
- computation is always local.

The Programming Language

Server

```
def AppletS (self) =
    self ? {
        applet_1(p) =
            p?(x)= P_1  |  AppletS[self],
        ...
        applet_k(p) =
            p?(x)= P_k  |  AppletS[self]
    }
in export applets
in AppletS[applets]
```

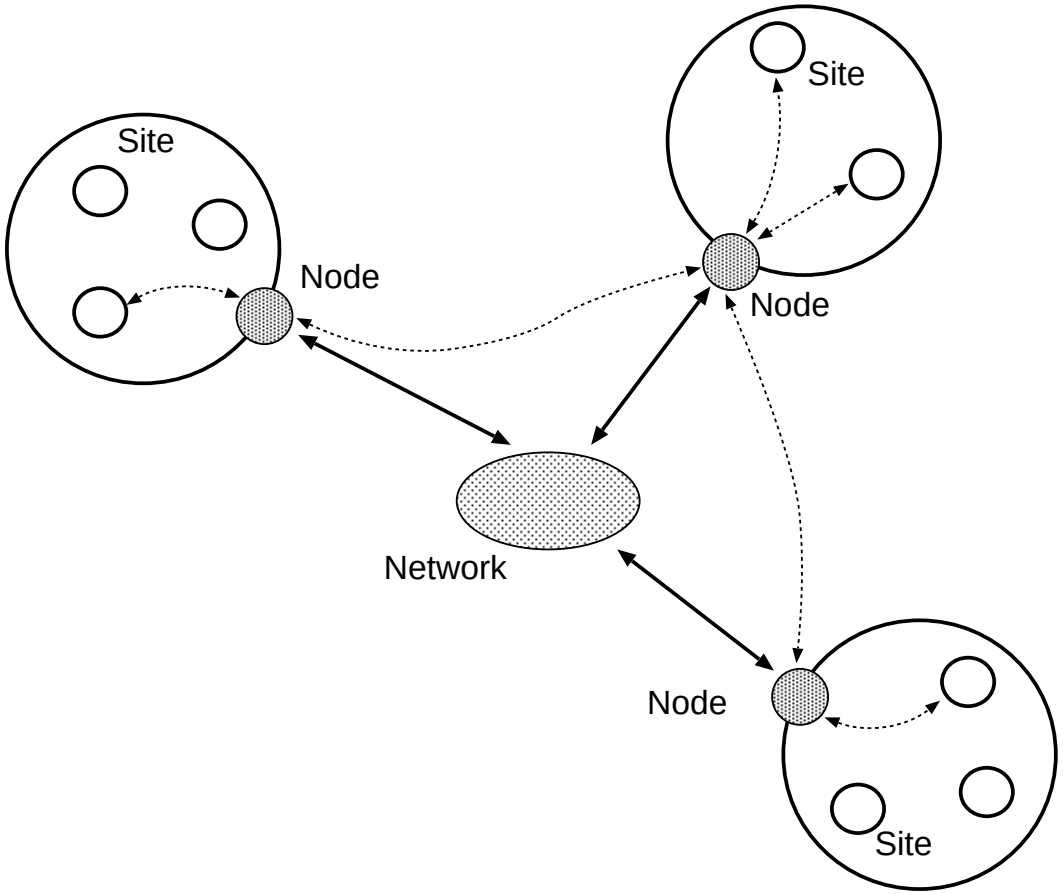
Client

```
import applets from Server
in new p applets!applet_j[p]  |  p![v]
```

The Computational Model

- flat network topology;
- peer-to-peer architecture;
- from a logical point of view we have just:
  - sites
- from an implementation point of view we have a three layered model:
  - network (provides a name-space);
  - nodes (proxies for sites);
  - sites.
- network and nodes do not compute;
- all computation is local to sites.

The Computational Model



The Computational Model • Software Organization

The Network

- **Interface NameSpace** – the name-space functionality;
- **Class Network implements NameSpace** – the network provides a name-space;

The Nodes

- **Interface Proxy** – the node manager for Distributed TyCO;
- **Class Node implements Proxy** – a Distributed TyCO node is a proxy for sites;

The Computational Model • Software Organization

The Sites

- **Interface InstructionSet** – the core TyCO instruction set;
- **Class VirtualMachine implements InstructionSet** – an implementation;
- **Class Externals** – operations external to the core TyCO definition;

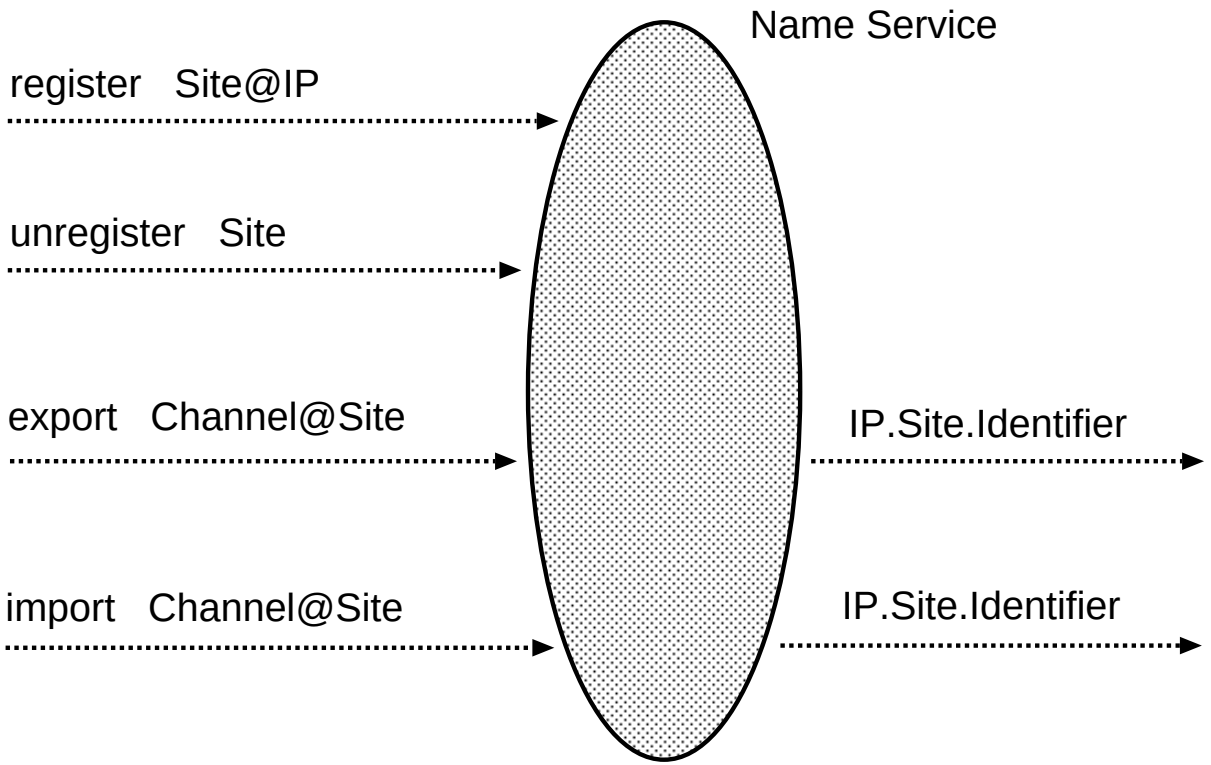
- **Class Site extends VirtualMachine**
  - I/O queues;
  - translation table;
  - operations supporting distribution and mobility implemented under the package Externals.

The Computational Model • The Network Layer

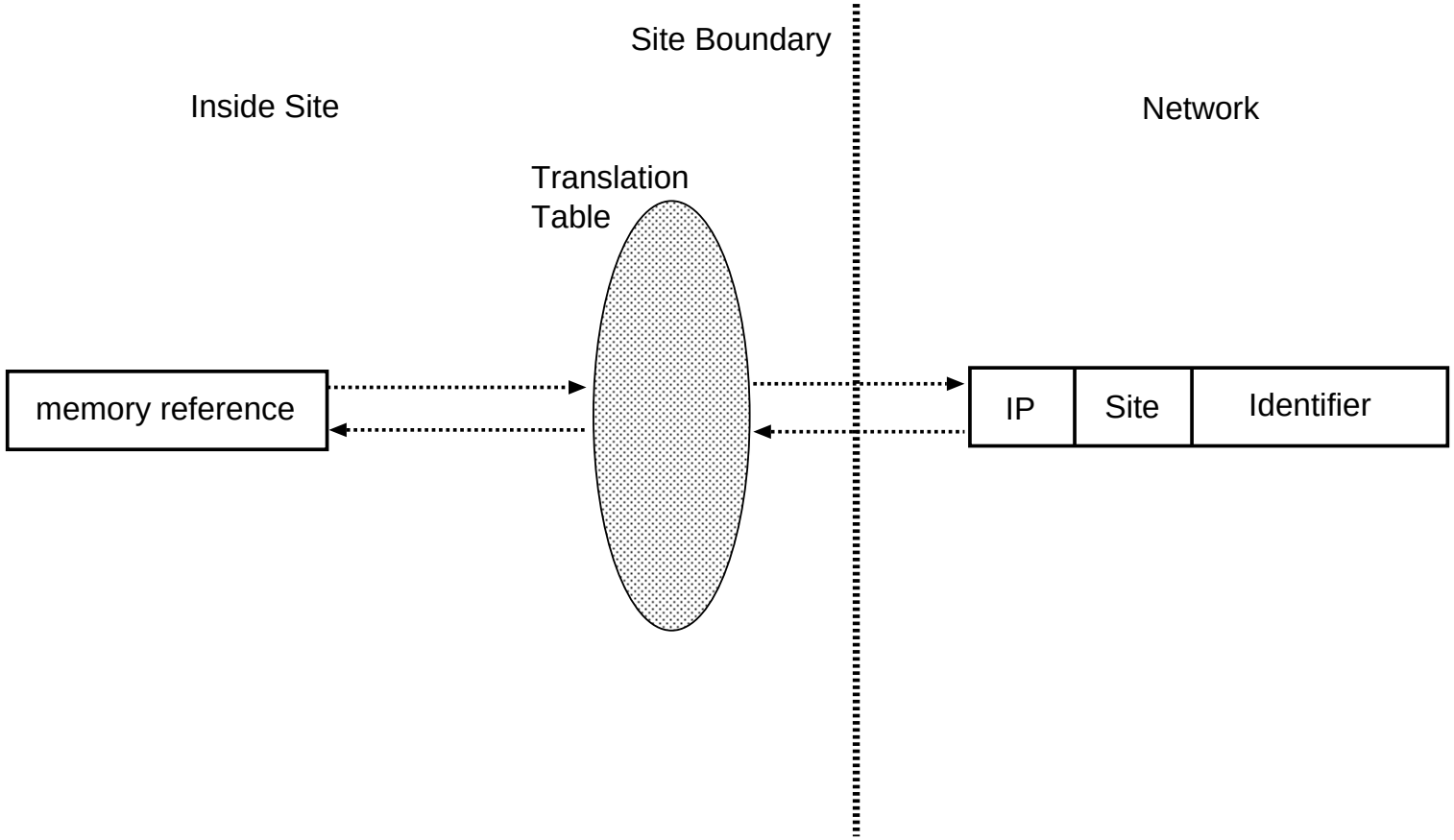
- network wide service that provides a name-space for computations;

- register a Site@IP;
- unregister a Site;
- register a resource Channel@Site;
- resolve a resource name Channel@Site.

The Computational Model • The Network Layer



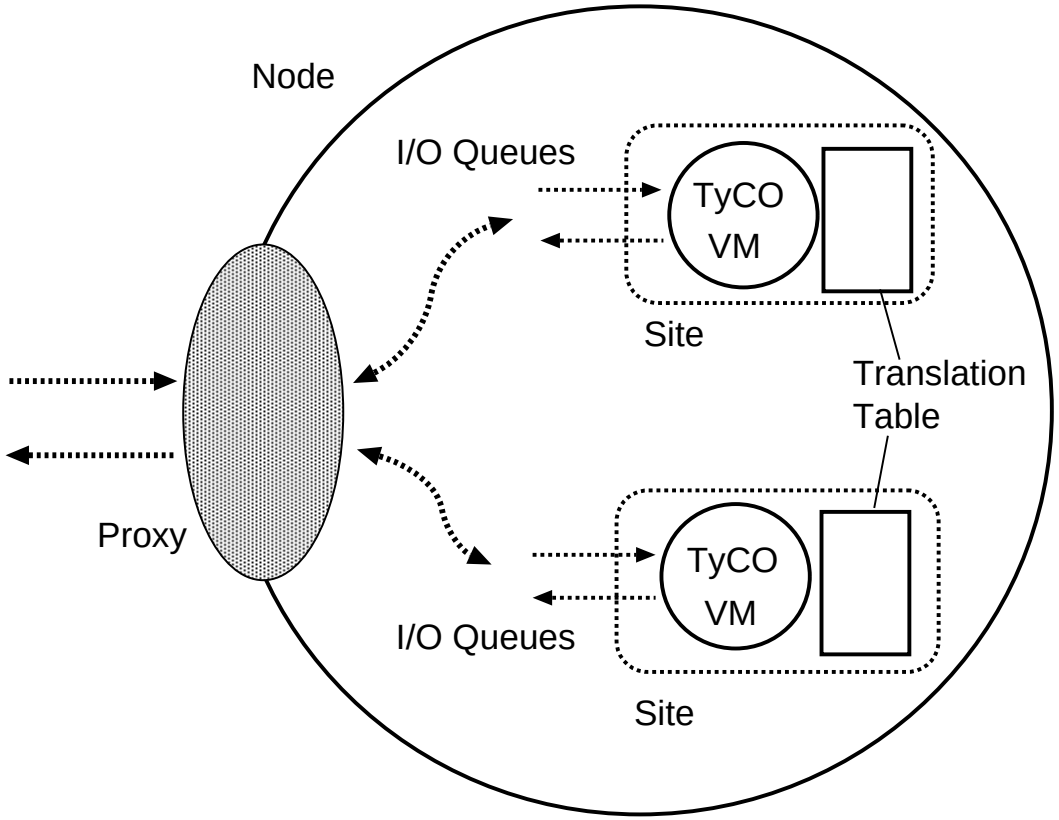
The Computational Model • The Network Layer



The Computational Model • The Node Layer

- one-to-one correspondence with physical/logical IP nodes;
- a proxy for sites that handles:
  - communication;
  - migration;
  - name-resolution;
- every IP node requires a Distributed TyCO proxy to run sites;
- the proxy shares an address space with the local sites;
- uses shared-memory for local communication.

The Computational Model • The Node Layer



The Computational Model • The Site Layer

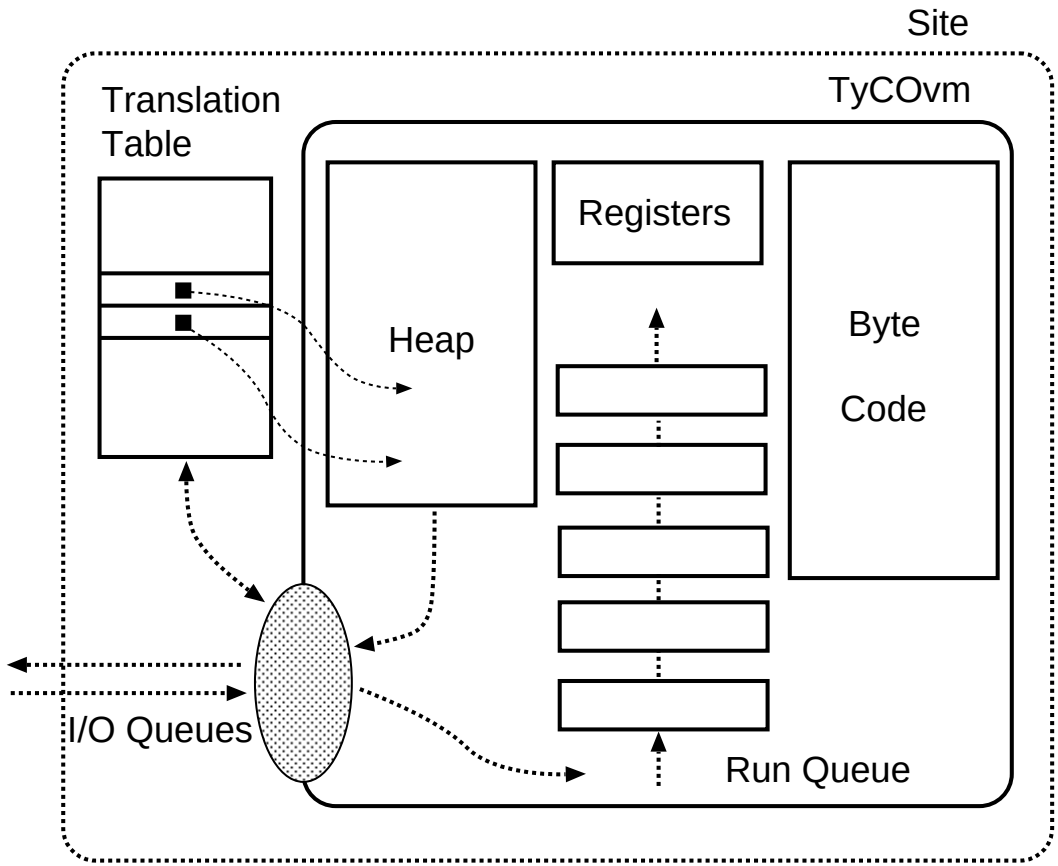
TyCO Virtual Machine

- register based virtual machine:
  - program area with byte code;
  - heap with activation records and channels;
  - general purpose registers;
  - run-queue for run-time generated tasks.
- single/multi-threaded.

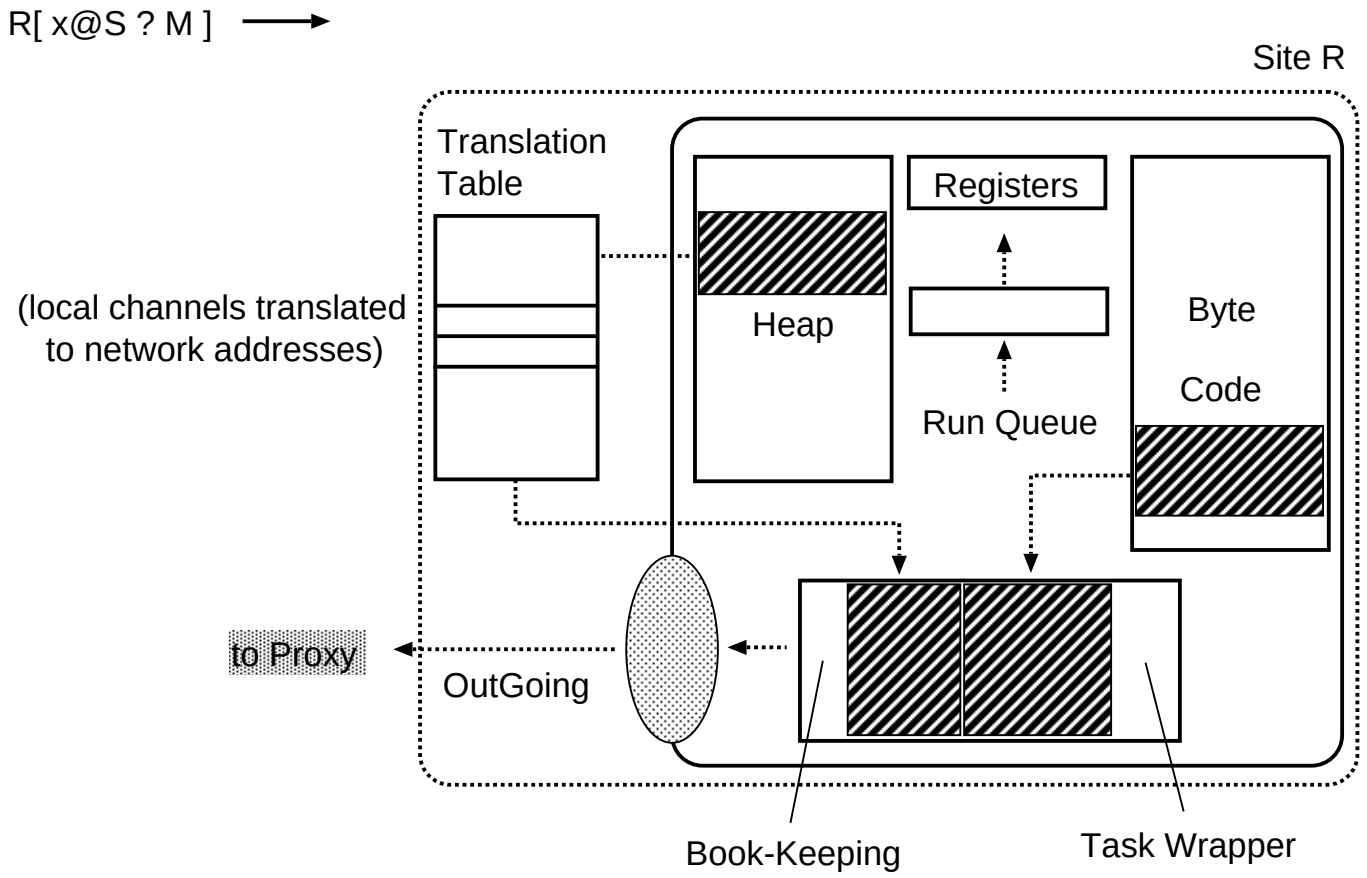
Extended with:

- queues for incomming/outgoing code;
- translation table for converting network references into local references;
- module of instructions for processing remote interactions.

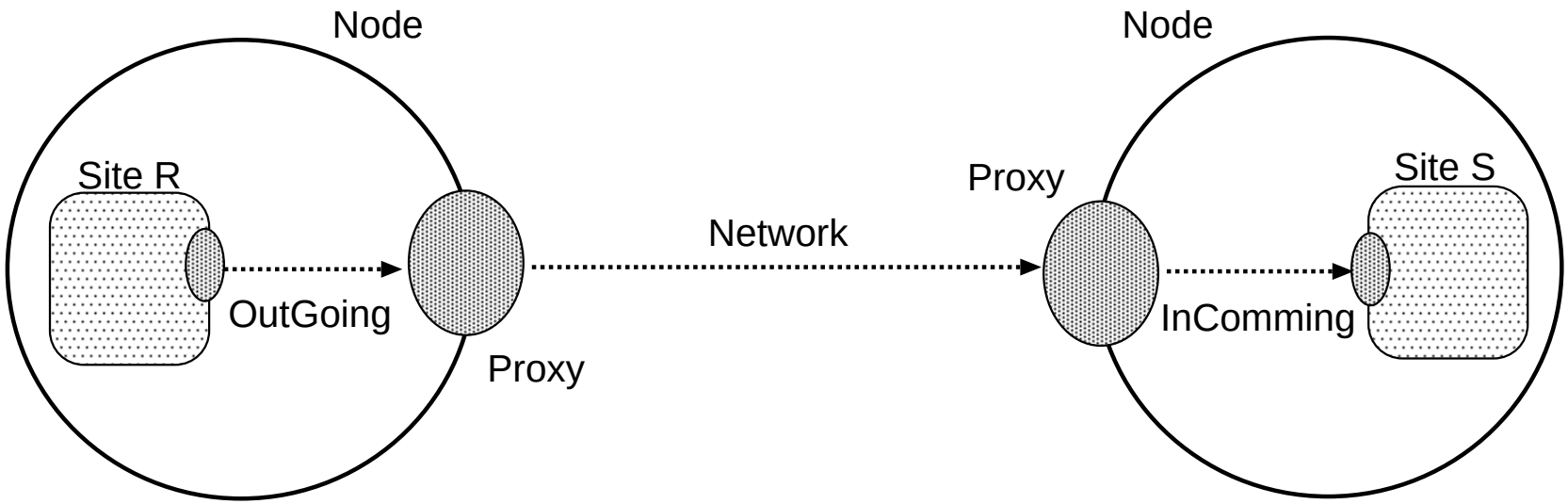
The Computational Model • The Site Layer



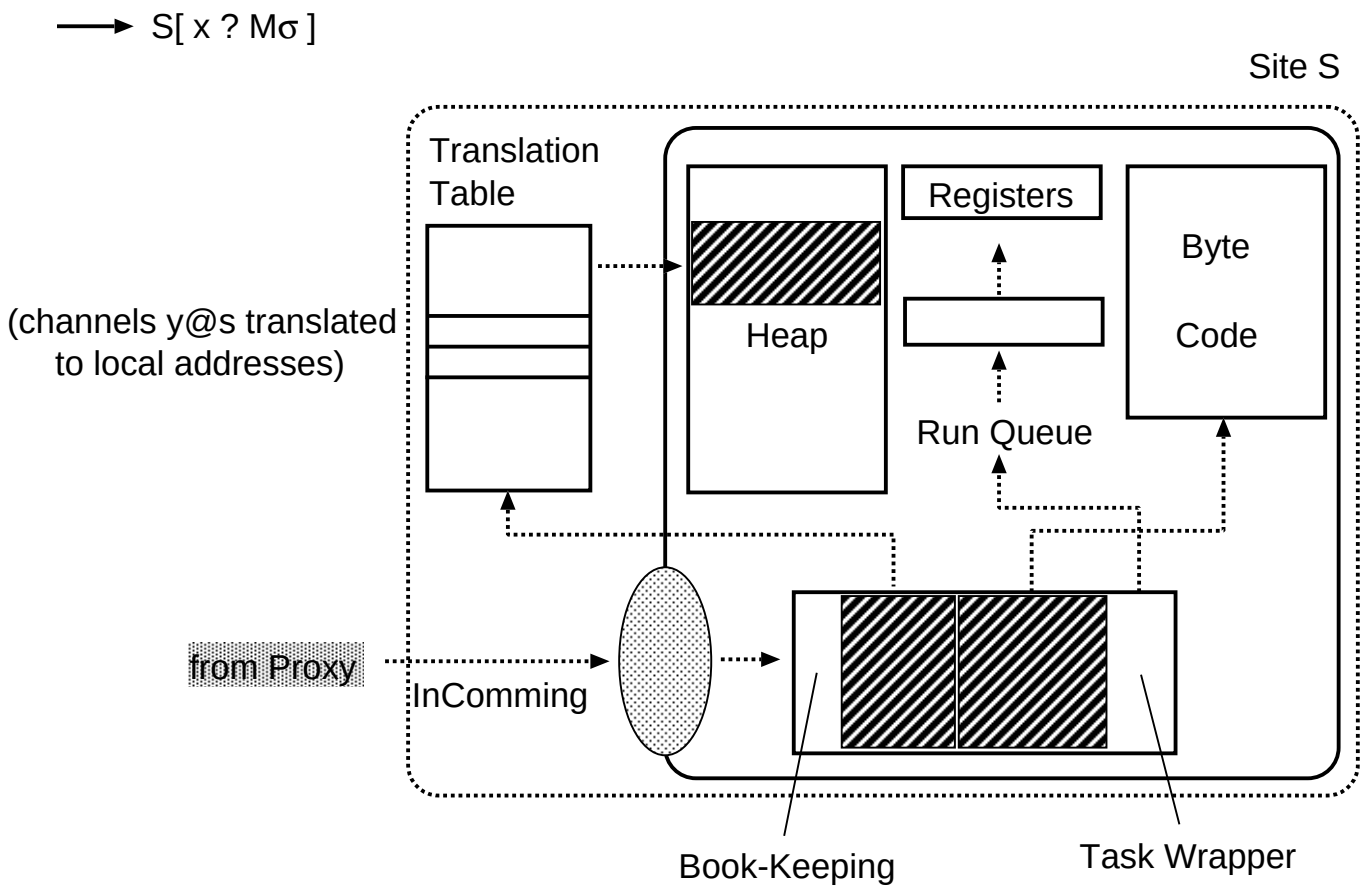
Computational Model • Putting it all Together



Computational Model • Putting it all Together



Computational Model • Putting it all Together



Current Work and Future Developments

- Preparation of next release of TyCO
  - major revisions of compiler and virtual machine;
  - infra-structure for distributed development;
  - system documentation;
  - open system.
- Implementation of Distributed TyCO in this environment
- Multithreaded Virtual Machine
- A Debugger for (Distributed) TyCO

URL $\Rightarrow$ `http://www.ncc.up.pt/~lblopes/tyco/`