Opportunities and Challenges of Global Computing

Rocco De Nicola

Dipartimento di Sistemi e Informatica
Università di Firenze

EU Project Consultation Meeting
Bridging Global Computing with GRID (BIGG)
Sophia Antipolis, France, November 28–29, 2006

Global Computers and Global Computing

Global Computer

Programmable open-ended computational infrastructure distributed at worldwide scale and available globally to provide uniform services with variable guarantees.

Global Computing

Global Computing refers to computation over a seamless, geographically distributed network of bounded resources by agents acting with partial knowledge and no central coordination exploiting their universal scale and the programmability of their services.

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);
- Web (client/server extended handshake);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);
- Web (client/server extended handshake);
- Telephone Network (guaranteed QoS);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);
- Web (client/server extended handshake);
- Telephone Network (guaranteed QoS);
- GRID (sharing of computing power);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);
- Web (client/server extended handshake);
- Telephone Network (guaranteed QoS);
- GRID (sharing of computing power);
- Ubiquitous Computing (seamless ubiquitous mobility);

Global Computers

There are many kinds of Global Computers:

- Internet (reliable stream transport);
- Virtual Private Networks (privacy and confidentiality);
- Web (client/server extended handshake);
- Telephone Network (guaranteed QoS);
- GRID (sharing of computing power);
- Ubiquitous Computing (seamless ubiquitous mobility);
- Diffused Computing (minutely diffused computational capacity).

Global Computers and Global Computing

Three distinguishing features

- Global Communication
  Different from, e.g., Send/Receive.

Global Computers and Global Computing

Three distinguishing features

- Global Communication
  Different from, e.g., Send/Receive.
- Global Computation
  Different from, e.g., Remote Procedure Call

Global Computers and Global Computing

Three distinguishing features

- Global Communication
  Different from, e.g., Send/Receive.
- Global Computation
  Different from, e.g., Remote Procedure Call
- Global Data
  Different from, e.g., Arrays and Records

Global Communication

- Hardwired communication: Whoever is capable of communication (holds one end of the wire) is always able to communicate (send/receive on the wire); unless, of course, something is broken; the delay is negligible.

Global Communication

- Hardwired communication: Whoever is capable of communication (holds one end of the wire) is always able to communicate (send/receive on the wire); unless, of course, something is broken; the delay is negligible.

- Global Communication:The capability to communicate does not mean being able to communicate.
  Wiring is very complicated; even when nothing breaks, things don't work, this might be due to:
  - Congestion (server not reachable),
  - Obstruction (obstacle for wireless),
  - Geography (No cell present),
  - Security (Firewall),
  - Privacy (No reply),
  - Physics (Long delays)
  - . . . .

Global Computation

How do we embed the features and restrictions of global communication in a computational model?

- It is not possible to use the familiar notion of function call/handshake and of pointers/references. There are no references that are always connected to their target, reconnection has to be considered.

Global Computation

How do we embed the features and restrictions of global communication in a computational model?

- It is not possible to use the familiar notion of function call/handshake and of pointers/references. There are no references that are always connected to their target, reconnection has to be considered.

- It is not affordable to have every function call over the network to block waiting for an answer and we cannot rely on the familiar notion of symmetric multi–party party (even async) channel communication. Consensus problems might be not solvable all the time.

Global Computation

How do we embed the features and restrictions of global communication in a computational model?

- It is not possible to use the familiar notion of function call/handshake and of pointers/references. There are no references that are always connected to their target, reconnection has to be considered.
- It is not affordable to have every function call over the network to block waiting for an answer and we cannot rely on the familiar notion of symmetric multi–party party (even async) channel communication. Consensus problems might be not solvable all the time.
- The familiar failure models must be abandoned and it cannot be assumed that every failure leads to an exception. It cannot even be assumed knowing that a failure happened.

Global Data

Semi-structured Data (XML) are important as universal standard for inter-operability of programming languages, databases, e–commerce, . . .

- No more arrays/lists, but tree structured data: Many children with the same label, instead of indexed children, mixture of repeated and non repeated labels under a node. New flexible type theories are required based on the effects of processes over trees or tree automata.

Global Data

Semi-structured Data (XML) are important as universal standard for inter-operability of programming languages, databases, e–commerce, . . .

- No more arrays/lists, but tree structured data: Many children with the same label, instead of indexed children, mixture of repeated and non repeated labels under a node. New flexible type theories are required based on the effects of processes over trees or tree automata.
- We cannot rely on uniform structure of data and need to abandon schemas based on records and disjoint unions and adopt self–describing data structures, e.g., edge–labeled trees (or graphs).

Global Data

Semi-structured Data (XML) are important as universal standard for inter-operability of programming languages, databases, e–commerce, . . .

- No more arrays/lists, but tree structured data: Many children with the same label, instead of indexed children, mixture of repeated and non repeated labels under a node. New flexible type theories are required based on the effects of processes over trees or tree automata.
- We cannot rely on uniform structure of data and need to abandon schemas based on records and disjoint unions and adopt self–describing data structures, e.g., edge–labeled trees (or graphs).
- A target would be a new, uniform, model of data and a new, uniform, model of computation on the Web, with opportunities for cross fertilization:

Global Data

Semi-structured Data (XML) are important as universal standard for inter-operability of programming languages, databases, e–commerce, ...

- No more arrays/lists, but tree structured data: Many children with the same label, instead of indexed children, mixture of repeated and non repeated labels under a node. New flexible type theories are required based on the effects of processes over trees or tree automata.
- We cannot rely on uniform structure of data and need to abandon schemas based on records and disjoint unions and adopt self–describing data structures, e.g., edge–labeled trees (or graphs).
- A target would be a new, uniform, model of data and a new, uniform, model of computation on the Web, with opportunities for cross fertilization:
  - Programming technology to typecheck, navigate, and transform both dynamic network structures and the semistructured data they contain.
  - Database technology to search through both dynamic network structures and the semistructured they contain.

Merging Global Computers

Overlay Computers

- Are abstractions that can be implemented on top of a global computer to yield yet other global computers.
- Represent families of potential, or actual, global computers by abstracting over common characteristics.

Merging Global Computers

Overlay Computers

- Are abstractions that can be implemented on top of a global computer to yield yet other global computers.
- Represent families of potential, or actual, global computers by abstracting over common characteristics.

Examples of Overlay Computers

1. resource discovery services (resource sharing in distributed networks);
2. search engines (abstraction of information repository);
3. systems of trusted mobile agents (autonomic, exploratory behaviour); IPSEC (abstraction of secrecy on IP);

Merging Global Computers

Overlay Computers

- Are abstractions that can be implemented on top of a global computer to yield yet other global computers.
- Represent families of potential, or actual, global computers by abstracting over common characteristics.

Examples of Overlay Computers

1. resource discovery services (resource sharing in distributed networks);
2. search engines (abstraction of information repository);
3. systems of trusted mobile agents (autonomic, exploratory behaviour); IPSEC (abstraction of secrecy on IP);

Non-Examples of Overlay Computers

1. Abstractions relying on: synchrony, low latency, limitless bandwidth,
2. Approaches neglecting scalability, security, quality of services.

Aims of research in Global Computing

Research on Global Computing, started roughly 10 years ago, but built on 40 years of research on concurrent programming, on theories, models and algorithms for distributed systems, and on networking has had the following main aims:

- identifying the class of global computers making sure that such a class is not unnecessarily restrictive;
- providing both foundational and practical advances on suitably large classes of global computers;
- integrating methods and concepts needed to understand general principles;
- identifying applications that can benefit from the general view and roam over global computers.

Clusters in GC1

A: Foundations of networks and large distributed environments:
Algorithmic aspects of distributed systems, dynamic societies of computational entities.

B: Analysis of Systems and Security:
Formal models and techniques for the specification of GC systems and the analysis of their behaviour.

C: Languages and programming environments:
Paradigms and tools for programming of GC systems, including the aspects of mobility, distribution, reconfigurability, security, and trust.

A: Foundations of Networks and large Distributed Environments

This cluster was meant to lay the foundations of global computing infrastructures. Its projects were dealing with issues such as scheduling algorithms and mechanism design for resource sharing between autonomously interacting agents, use of type-theoretic approaches for the specification and verification of distributed computing systems, and other paradigms related to the reasoning about properties of such systems.

1. Critical Resource Sharing for Cooperation in Complex Systems

CRESSCO: Conducted research on foundational aspects of managing critical resources (e.g. bandwidth, frequency, energy, processor time) in GC infrastructures connecting very large numbers of independent, possibly mobile and selfish agent entities.

A: Foundations of Networks .... ctd.

2. A Data-Centric Approach to Global Computing

DBGLOBE: Aimed at extending current database technology in order to address data management requirements in large-scale networks of mobile entities using a data-centric and service-oriented approach to GC.

3. Foundational Aspects of Global Computing Systems

FLAGS: Had the objective of providing a general set of design principles and mechanisms facilitating the construction of global computing systems with autonomous and selfish agents competing for resources.

4. Societies of Computees

SOCS: Investigated computational and logical models for describing, analyzing, and verifying properties of individual agents as well as of societies of agents. The main focus was on formal models that identify core functionalities that the entities need to have in a GC environment.

B: Analysis of Systems

The focus of this cluster was on the use of formal techniques such as type theory, proof carrying code, and formal models for trust management in order to validate system properties such as correctness and security.

1. Dynamic Assembly, Reconfiguration and Type Theory

DART: Developed formalisms and techniques for the modeling of the temporal dimension of GC systems in order to support arbitrary interleaving of type-checking, meta-programming, and normal computational activities while retaining safety.

2. Mobile Resource Guarantees

MRG: Extended the concept of Proof-Carrying Code (PCC) to include guarantees of resource usage, both in high-level source code, i.e. in a user-oriented programming language, and in low-level target code, e.g. a virtual machine's byte code.

B: Analysis of Systems ctd.

3. Models and Types for Security in Mobile Distributed Systems

MYTHS: Explored and developed type-based theories of security for mobile and distributed systems.

4. Proofs of Functionality for Mobile Distributed Systems

PROFUNDIS: Conducted research on formal modelling and verification techniques to explore key issues such as security protocols, authentication, access rights and resource managemen. Automatic support for the design and implementation of such features was provided.

5. Secure Environment for Collaboration among Ubiquitous Roaming Entities

SECURE: Developed a formal model of security based on the notion of trust and algorithms for the dynamic and self-configuring management of trust.

C: Languages and Programming Environments

This third cluster has used and extended the theoreies developed and/or considered in the in the other two clusters to come up with software tools and frameworks for building global computing infrastructures.

1. Architectures for Mobility

AGILE: Explored an UML-based architectural approach to software engineering for mobile systems based on a uniform mathematical framework. This includes theoretical foundations as well as pragmatic techniques for designing mobile computing systems by focusing on software architecture and corresponding UML models.

2. Design Environments for Global Applications

DEGAS: Proposed methods for deriving models such as process algebras and performance models from annotated high-level application models in UML to support the design and implementation of global computing applications.

C: Languages and Programming Environments ctd.

3. Mobile Calculi Based on Domains

MIKADO: Developed new formal programming models for global computing based on the concept of process mobility and the notion of "domain". The formal models were used to build new prototype programming languages and runtime environments to implement new calculi for GC.

4. Peer-to-Peer Implementation and Theory

PEPITO: Investigated the foundations of scalable distributed computing based on the peer-to-peer (P2P) computing paradigm. P2P algorithms, programming language features were explored and a generic, language-independent distribution platform for P2P computing was developed.

Criticisms:

The reviews of the GC1 projects put forward three main general criticisms that applied to different extents to the different projects:

1. Limited connections to applications, due to limited attention devoted to everyday practice;
2. Limited development of middleware that hindered the possibility of validating the quality of the developed frameworks;
3. Lack of an horizontal dimension between the different global computers.

Global Computing II

Key aim of GC2 initiative

The aim of the second edition that obviously builds on the first one, is to reach a substantial integration between theory, systems building and experimentation.

Long terms expected results

Achieve real, integrated global computing in a wide range of application scenarios by providing foundational advances on suitably large classes of global computers, together with the integration of methods, concepts and tools.

Focus of the research

Focus on common features representing a family of potential or actual global computers described by appropriate abstractions (overlay computers) to yield enhanced classes of global computers that are programmable and computationally complete in their application domain.

Research Themes for Global Computing

Security

- Authentication, privacy, non-repudiation, authentication, anonymity, secure distributed computation;
- Methods and infrastructures for trust, and trust formation and management;
- Resource Usage and Protection of resource bounds (CPU, disk space, bandwidth, information/application level resources, . . .).

Computational Models

- Models of interaction: cooperation and interoperability;
- Components and modularity; wireless devices with limited computational power, limited availability;
- Abstraction mechanisms, programming languages concepts;
- Possibility of distribution transparency.

Research Themes for Global Computing

Network Awareness

- Quality of Services;
- Autonomy, adaptivity, and self-organization;
- Ad hoc assemblies of computees;
- Dynamic learning about environment and peers;
- Scalability.

Support Tools

- Validation and verification;
- Algorithmic principles;
- Design support and software techniques.

Mobius : Mobility, Ubiquity, Security

Coordinator INRIA Sophia Antipolis (France)

OBJECTIVES

Establish a security architecture appropriate for global computers:

- Adopt a computational model that captures faithfully fundamental aspects of global computers,
- Identify the trust and security requirements of such a model,
- Develop on top of the computational model a security framework that enforces these requirements,
- Provide the enabling technologies necessary for implementing the framework,
- Validate the architecture.

Mobius Technologies and Models

Mobius Computational Models

Very large networks of JVM-enabled devices:

- Flexibility and uniformity: aimed at providing a global and uniform access to services,
- Security and heterogeneity: subject to resource and security constraints

Mobius Basic Technologies

Precision and automation are needed to guarantee applicability and scalability. Automatic and interactive technologies are combined:

- Type systems: Efficient, automatic, but specialized and imprecise. Used for information flow, resource usage, aliasing.
- Program logics: general, precise, but interactive. Used for the characterization of non-functional properties, of high-level security policies of component correctness.

Mobius Security Issues

Securing very large networks of JVM-enabled devices:

- Devices are protected individually by means of static and resource aware enforcement mechanisms,
- No sharp distinction between static Trusted Computing Base and mobile applications,
- No central trust authority: trust infrastructures must allow verifiable evidence (cryptography is not enough),
- Expressive security policies and functional specifications.
- Security framework with appropriate characteristics based on ideas from Proof Carrying Code (PCC)
- downloaded components come equipped with certificates, i.e., condensed and formalized mathematical proofs, which are unforgeable and straightforward to check.

AEOLUS: Algorithmic Principles for Building Efficient Overlay Computers

Coordinator University of Patras (Grece)

OBJECTIVES

- To identify and study the important fundamental problems and functionalities of overlay computing and investigate the related algorithmic principles and tools for the programmers.
- To develop, rigorously analyze, and experimentally validate algorithmic methods that make functionalities efficient, scalable, fault-tolerant, and transparent to heterogeneity.
- To provide improved methods for communication and computing among wireless , possibly mobile, nodes so that they can transparently become part of larger global computers.
- To implement a set of functionalities and integrate them under a common software platform to provide the basic primitives of an

AEOLUS: challenges

Efficiency and scalability are fundamental requirements to guarantee success of global systems. New issues have to be faced due to: Scale, heterogeneity, high dynamicity, diversity in ownership of resources, selfishness, untrustworthiness, limited knowledge, fairness vs efficiency.

Expected Results

- Formulation of algorithmic principles for overlay computers aiming at understanding of the models, designing provably efficient algorithms, statements of trade-offs, impossibility results and lower bounds;
- Implementation of functionalities dealing with secure data mining, distributed trust management, selfishness/maliciousness, coping with imperfect/partial/uncertain knowledge;
- Integration of functionalities into a common platform to serve as the General-Purpose Programmable Overlay Computer;
- Proof-of concept application

SENSORIA: Software Engineering for Service-Oriented Overlay Computers

Coordinator LMU Munich (Germany)

OBJECTIVES

- To develop a novel comprehensive approach to Engineering of software systems for Service-Oriented Overlay Computers integrating
  - foundational theories (process calculi, type theory, stochastic mod.)
  - techniques, methods and pragmatic of software engineering for services
  - software tools to support services development, assessment, deployment and management.
- Investigate the impact of the developed approach on a number of application areas:
  - e-business,
  - automotive systems,
  - e-learning,
  - telecommunications

SENSORIA Challenges

E-Services

Selling services has become the biggest growth business in the IT industry. Service-oriented computing is becoming the driving force behind innovation in IT-industry. Competitiveness of European industries depends on early and successful adoption of this new paradigm. ...
Europe lags behind both in B2B and in B2C (a Microsoft research leader says that Europe is at least three years dog behind US)

Service Development

- Service-Oriented Computing has been addressed by IT industry only in an ad-hoc and undisciplined way.
- Applications have the ability to "talk" to each other but they do not "understand" what they are talking about (only syntactic conformance),
- Web service standards have poor semantic foundations

SENSORIA

Expected Results

- General concept of services that is independent from particular global computer and from any programming language: Services as autonomous, platform-independent computational entity that can be described, published, categorised, discovered
- Services described in a modular way, so that security issues, quality of service measures and behavioural guarantees are preserved under composition.
- Languages and models for global service-oriented systems with full mathematical semantics equipped with theories and methodologies for qualitative and quantitative analysis (quality of service, security, performance, resource usage, . . .).
- Sound engineering methods and deployment techniques relying on model-based transformations. Re-engineering of legacy systems. Validation via a number of case studies.

Many thanks for your attention