Autonomic Computing

l'Autogestion de Systèmes Informatiques

Ada Diaconescu

ada.diaconescu@telecom-paristech.fr

les Systèmes Informatiques aujourd'hui

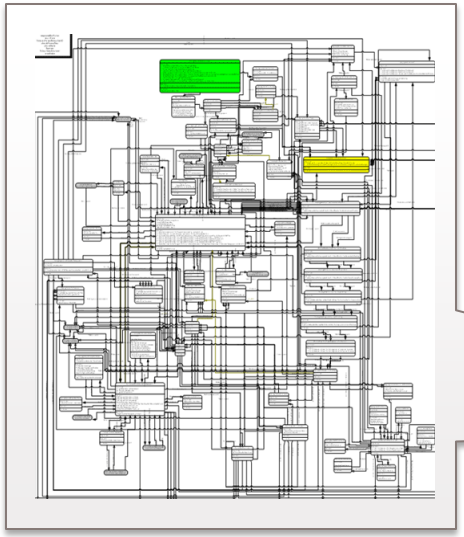
- **Essentiels dans des nombreux domaines du monde réel**
  - Commerce, opérations bancaires, transport, médecine, sécurité, communication et multimédia, …
  - Exemple de domaine qui ne soit pas affecté par l'informatique?

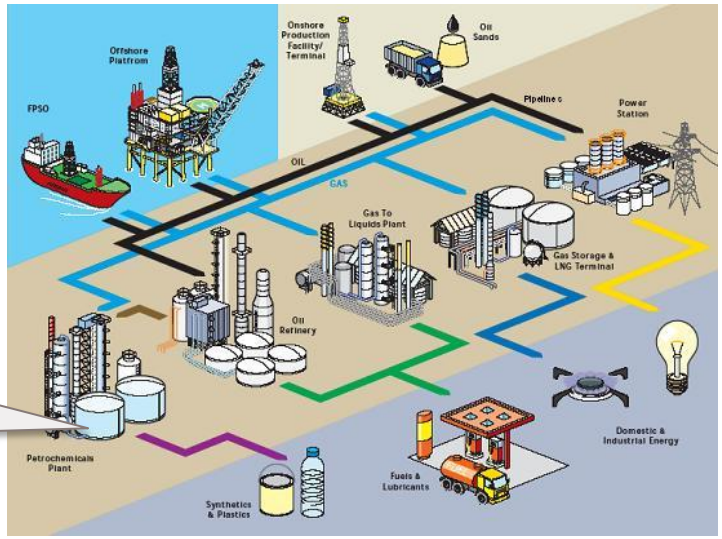
- **De plus en plus complexes**
  - Compliqués, répartis, hétérogènes, interconnectés et dynamiques
  - Même si on métrise chaque composant ou application séparément, le système résultant par l'intégration des multiples composants ou applications devient difficile à gérer - « More is different »

- **Besoins d'extensibilité et d'adaptabilité statique et dynamique**
  - Pressions du marché
  - Evolution du contexte d'exécution

l'Importance des Systèmes Informatiques

- L'informatique permet l'automatisation de processus / routines
- Evolution de tâches de plus en plus avancées
- Un des effets secondaires inévitable: la complexité





le Défi

- **La complexité de systèmes informatiques devient de plus en plus difficile à gérer à la main**
  - Coûts élevées
  - Erreurs fréquentes
  - Manque d'experts
  - Temps de réaction insuffisants
- **Les systèmes informatiques sont mises en difficulté par leur propre succès**

> "Dealing with <complexity> is the single most important challenge facing the I/T industry. It is our next Grand Challenge."
>
> P. Horn, IBM manifesto, October 2001

L'Importance de l'Autogestion

"Civilization advances by extending the number of important operations which we can perform without thinking about them."

Alfred North Whitehead

- Exemple: La crise de la Téléphonie en 1920
  - Cause : l'expansion rapide de l'utilisation du téléphone
  - Conséquence : les analystes avaient prédit que par 1980 chaque femme dans les Etats Unis devrait travailler en tant que standardiste téléphonique si la croissance continuait au même rythme
  - Besoin : automatiser les standards téléphoniques manuels
- Nous somme confrontés à une crise similaire dans le domaine de l'informatique

L'Importance de l'Autogestion Informatique

". . . incredible progress in almost every aspect of computing — microprocessor power up by a factor of 10,000, storage capacity by a factor of 45,000, communication speeds by a factor of 1,000,000 — but at a price.

Along with that growth has come increasingly sophisticated architectures governed by software whose complexity now routinely demands tens of millions of lines of code. . . .

Even if we could somehow come up with enough skilled people, the complexity is growing beyond human ability to manage it. As computing evolves, the overlapping connections, dependencies, and interacting applications call for administrative decision making and responses faster than any human can deliver.

Pinpointing root causes of failures becomes more difficult, while finding ways of increasing system efficiency generates problems with more variables than any human can hope to solve."

[IBM Corporation, 2001]

La solution de l'Autonomic Computing

- **Rendre les systèmes informatiques capables de s'autogérer**
  - Solution inspiré par le système nerveux autonome humain
    - Développe des stratégies et des algorithmes pour la gestion de la complexité
    - Gere la manque d'information
  - Initiative IBM, 2001

- **Minimiser le besoin d'intervention humaine**
  - Processus de gestion autonome, guidés par des politiques de haut niveau spécifiées par l'humain

L'Autogestion Informatique vue par IBM

"It's time to design and build computing systems capable of running themselves, adjusting to varying circumstances, and preparing their resources to handle most efficiently the workloads we put upon them. These autonomic systems must anticipate needs and allow users to concentrate on what they want to accomplish rather than figuring how to rig the computing systems to get them there. . . .

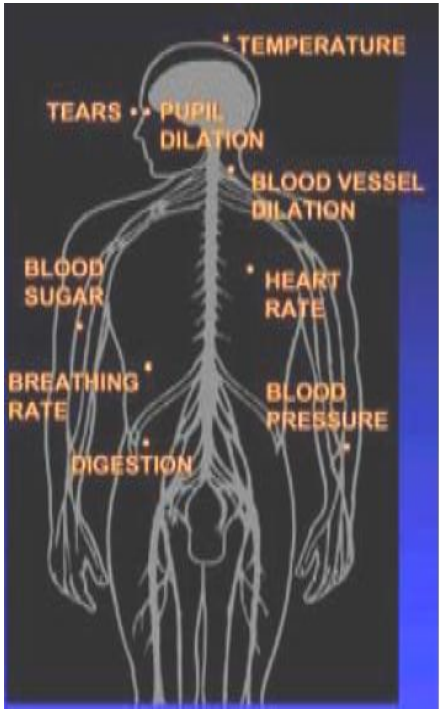
It is the self-governing operation of the entire system, and not just parts of it, that delivers the ultimate benefit."

[IBM Corporation, 2001]

Inspiration pour l'Autogestion Informatique

"What is autonomic computing? It is the ability of systems to be more self-managing. The term autonomic comes from the autonomic nervous system, which controls many organs and muscles in the human body. Usually, we are unaware of its workings because it functions in an involuntary, reflexive manner -- for example, we don't notice when our heart beats faster or our blood vessels change size in response to temperature, posture, food intake, stressful experiences and other changes to which we're exposed. And, by the way, our autonomic nervous system is always working."

Alan Ganek, VP Autonomic Computing, IBM



les Avantages de l'Autogestion

- **Avantages immédiates ou à court terme :**
  - Réduire les coûts
    - Coûts de gestion - minimiser la dépendance sur l'intervention humaine
    - Coûts d'utilisation de ressources - logiciels, matériels et consommation de courant
      - Ex: optimiser les traitements de données, la communication, l'utilisation du stockage, l'utilisation de CPUs en attente (grid et cloud computing), …
  - Assurer une meilleure expérience utilisateur
    - Systèmes informatiques plus réactifs et plus simples à utiliser
    - Systèmes informatiques plus fiables, disponibles et sécurisés
- **Avantages à longue terme**
  - Etre capable de construire des systèmes informatiques de plus en plus complexes, à base des briques existants, fiables et performants
  - Permettre aux non-experts d'utiliser et/ou de développer de systèmes
  - …

Fonctions Principales

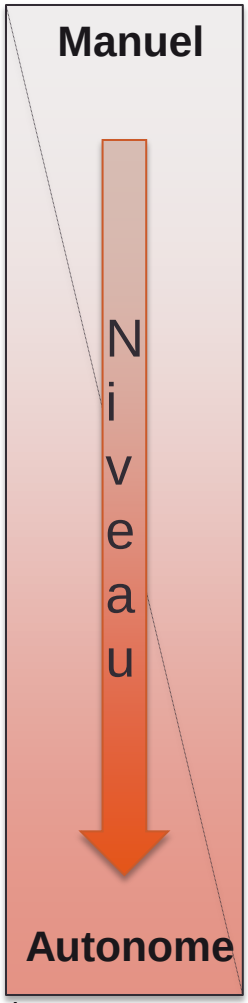
- **Auto-optimisation**
  - Détecter la dégradation de performance du système
  - Prendre des actions qui améliorent la performance du système
- **Auto-réparation**
  - Détecter des problèmes
  - Se reconfigurer de façon à assurer le fonctionnement continu
- **Auto-configuration**
  - Ajuster ses ressources à l'exécution afin de s'adapter aux changements de son état interne et de son environnement externe
- **Auto-protection**
  - Détecter des attaques internes et externes et protéger ses ressources de façon à maintenir la sécurité et l'intégrité du système

Fonctions principales (+)

- **Sensitivité au contexte**
  - Avoir conscience de son environnement et être capable de réagir aux changements
- **« Conscience »**
  - Connaissance de soi-même : état et comportement
- **Anticipation**
  - Prévoir ses besoins et comportements ainsi que ceux de son environnement et être capable de s'autogérer de façon proactive
- **Ouverture**
  - Portable à travers plusieurs plates-formes logicielles et matérielles
  - Construit sur des interfaces et des protocoles ouverts et standards

l'Evolution vers l'Autogestion

- **1. Niveau élémentaire**
  - Plusieurs sources de données générées par le système
  - Besoin important d'intervention par des experts IT
- **2. Niveau géré**
  - Consolidation de données via des outils de gestion
  - Les administrateurs analysent les données et prennent des décisions
  - Le système a plus de connaissance de soi-même
  - Meilleure productivité
- **3. Niveau prédictive**
  - Le système collecte des données, établit des corrélations et propose des actions
  - Les administrateurs autorisent et initialisent les actions
  - Moins de dépendance sur les compétences des experts
  - Prise de décision plus efficace et plus pertinente
- **4. Niveau adaptative**
  - Le système collecte des données, établit des corrélations et prend des actions
  - Les administrateurs gèrent le rapport entre la performance du système et les contrats avec les clients (SLAs)
  - Système plus agiles et résilients, tout en demandant un minimum d'intervention humaine
- **5. Niveau autonomique**
  - Des composants intégrés, gérés pendant l'exécution en conformité avec des politiques de haut niveau
  - Les administrateurs se concentrent sur les besoins de « business »
  - Les politiques de « business » contrôlent la gestion du système
  - Du « business » plus agile et plus résilient



IBM Global Services and Autonomic Computing, IBM White Paper, October 2002; http://www-3.ibm.com/autonomic/pdfs/wp-igs-autonomic.pdf

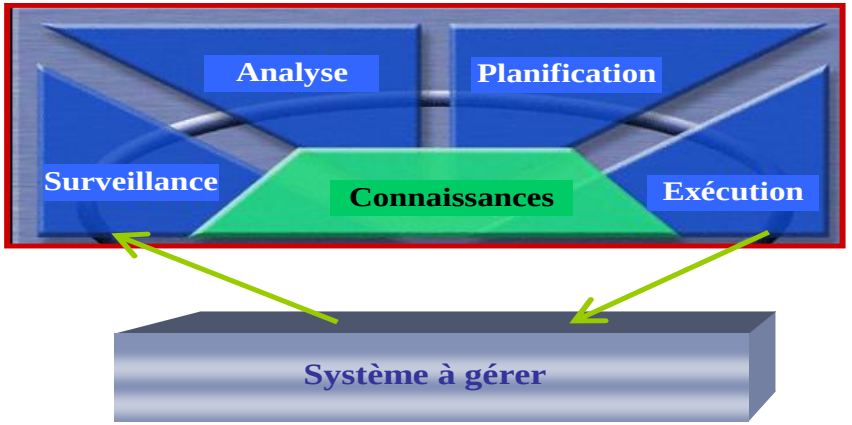
Etat courant

- **Domaine assez récent (~2000)**
- **S'appuie sur des travaux existantes**
  - Logiciels auto-adaptables, robotique, automatique, IA, …
- **Mais, beaucoup de défis importants à surmonter!**
  - Domaine de recherche en plein croissance
- **Résultats disponibles**
  - Industrie: quelques outils et facilités technologiques disponibles
    - L'humain reste dans la boucle
    - Quelques fonctions complètement automatisées (ex: dimensionnement de pools d'instances dans les serveurs d'application, distribution automatique de charge, ..)
  - Recherche: premiers canevas d'autogestion
    - Différents projets se concentre sur des différents aspects ou types de système
    - Besoin d'approfondir et d'intégrer des solutions partielles

Propriétés Attendues des Système Autonomes

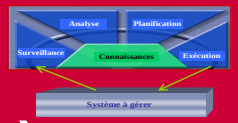
- Réactivité
- Performance
- Efficacité
- Fiabilité
- Robustesse
- Prédictibilité
- Disponibilité
- Confidentialité
- Sécurité
- …
- Comment garantir, comment éviter ?...

Architecture Générale



- Surveillance : obtenir des données sur le système géré
- Analyse : détecter les problèmes
- Planification : trouver des solutions
- Exécution : modifier le système géré
- Connaissances : représenter le contexte, le système, …

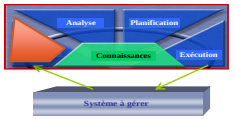
La boucle MAPE-K
(Monitoring, Analysis, Planning, Execution and Knowledge)



- **Comment implanter les différents fonctions d'une boucle ?**
  - Centres d'intérêt et approches différents selon
    - L'objectif de l'autogestion (ex: optimisation, réparation, configuration, protection, …)
    - La ressource géré (ex: service ou application logiciel; composant matériel; …)

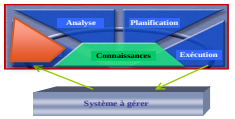
- **Comment intégrer les différents boucles autonomiques ?**
  - Différents objectifs d'autogestion
    - Ex: l'optimisation de la performance peut être en conflit avec la protection
  - Différents ressources gérés
    - Ex: l'optimisation individuelle de chaque service ne garantit pas l'optimisation globale d'une application à services
  - Différents niveaux d'abstraction
    - Ex: l'optimisation du système du point de vue de l'administrateur doit se traduire dans des mécanismes d'optimisation au niveau des ressources sous-jacentes

La Surveillance



- **Déclanchement – Quand ?**
- **Objectif – Pourquoi?**
  - Obtenir des données sur les ressources gérés et leurs environnement
    - Performance: temps de réponse, « throughput », consommation de ressources, …
    - Charge système: nombre de clients et leurs comportements
    - Fiabilité: périodes de fonctionnement « normal », pannes, exceptions, …
    - Architecture: composants opérationnels et leurs interconnections
    - Configuration: valeurs de paramètres
    - …
- **Moyens – Comment?**
  - Interception d'appels – méthode non-intrusive
  - Modification de code – méthode intrusive
  - …

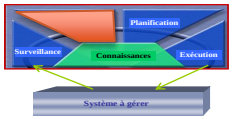
La Surveillance - Problématiques



- **Quels aspects faudrait-il surveiller ?**
  - Impossible de tout surveiller..
  - Avec quelle fréquence ?
  - Avec quels délais introduits ?
  - Surveillance adaptative?
  - ...
- **Comment obtenir les données ? – les capteurs**

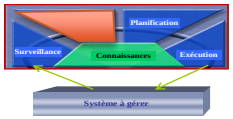
  Comment concevoir les systèmes afin de faciliter la surveillance ?
  - Interception d'appel de méthode
  - Surveillance d'attributs (ex: JMX)
  - Sondes système spécifiques (ex: CPU, mémoire, disque, …)
  - ...
- **Comment transmettre les données?**
  - Communication par événement ou message (publication/suscription, point-à-point)
  - Communication par appel de méthode
  - ...

L'Analyse



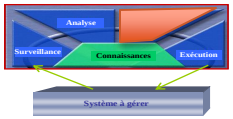
- **Déclanchement – Quand ?**
  - Lors de la disponibilité de nouvelles données de surveillance
  - Périodiquement, ou à des moment prédéfinis
- **Objectif – Pourquoi?**
  - Détecter (ou prédire) des problèmes, des symptômes ou des changements
    - Seuil dépassé (ex: surcharge d'une ressource)
    - Faute logiciel (ex: exception)
    - Panne matériel (ex: serveur ou connexion en panne)
    - Anti-patterns de performance dans l'architecture ou le comportement de l'application
    - ...
    - Changement de valeurs de variables
    - Changement de l'architecture ou du comportement
    - Augmentation dans l'usage de ressources
    - ...
- **Moyens – Comment?**
  - Utilise les données obtenues par la Surveillance et/ou disponibles d'auprès la base de connaissances
  - Algorithmes spécifiques d'analyse de données

L'Analyse - Problématiques



- **Comment détecter des problèmes complexes ?**
  - Corrélation de données provenant de plusieurs sources
- **Comment gérer l'incertitude ?**
  - Signaler les problèmes potentiels ou prédits?
- **Comment gérer ses ressources ?**
  - Quand faut-il utiliser des algorithmes coûteux?
- **Comment temporiser les signalisations ?**
  - Signaler un problème importante le plus vite possible (ex: dès le premier symptôme), ou s'assurer d'abord de son existence (en risquant une signalisation tardive)
  - Continuer à signaler / supprimer / retarder les problèmes mineurs dans la présence d'un problème majeure?
- **…**

La Planification

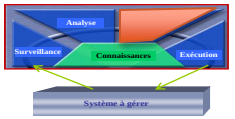


- **Déclenchement – Quand?**
  - Par l'analyseur, lors de l'identification d'un symptôme ou d'un problème
  - Périodiquement, ou à des moments prédéfinis

- **Objectif – Pourquoi?**
  - Trouver des solutions aux problèmes identifiés par l'analyseur

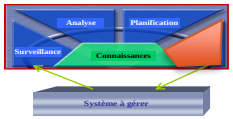
- **Moyens – Comment?**
  - Logique de planification - politiques décisionnelles
    - Prédéfinies
    - Evolutives (ex: apprentissage automatique)
  - Connaissances
    - Modèles de l'application et de son environnement
    - Historique de plans précédents et de leurs résultats
    - …

La Planification - Problématiques



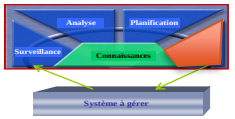
- **Comment exprimer la logique de planification?**
  - Règles ECA (événement, condition, action)
  - Règles orientées objectif (« goal-oriented rules»)
  - Règles guidées par l'utilité (« utility-based rules»)
- **Comment faire face à l'incertitude?**
  - Manque d'information, changements et évolutions
    - Architecture et comportement de l'application et de son environnement
    - Conséquences d'un changement à court et à longue terme
- **Comment gérer les conflits?**
  - Objectifs
    - ex: la performance versus la sécurité ou la fiabilité
  - Champ d'application
    - ex: optimisation locale versus optimisation globale
  - Niveau d'abstraction
    - ex: politique au niveau administrateur système versus mécanisme d'auto-configuration spécifique à une ressource géré
- ...

L'Exécution



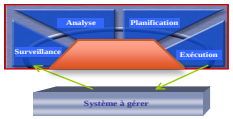
- **Déclenchement – Quand?**
  - Selon les solutions proposées ou imposées par la Planification
- **Objectif – Pourquoi?**
  - Modifier le système géré sans l'arrêter - actions possibles :
    - Reconfigurer des paramètres
    - Modifier l'implémentation
    - Changer des interconnections
    - Créer, détruire ou faire migrer des instances logiciels
    - Ajouter ou retirer des ressources matériels
    - ...
  - Modifier l'environnement d'exécution (effet indirect par le système géré)
    - Incrémenter ou décrémenter l'utilisation de ressources logiciels et/ou matériels
- **Moyens – Comment?**
  - Techniques spécifiques à chaque type de changement et de système

L'Exécution - Problématiques



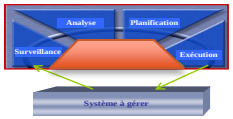
- **Comment concevoir les systèmes afin de les rendre modifiables dynamiquement ?**
  - Caractéristiques nécessaires de l'application
    - Interfaces d'accès aux paramètres
    - Points d'interception
    - Modularité
    - Faible couplage
    - ...
  - Technologies adaptées – souplesse, flexibilité, extensibilité
    - Modularité et séparation des préoccupations: composants, services, aspects
    - Faible couplage: communication par événement/message (publication/suscription, bus)
- **Comment assurer la qualité de service suite aux changements dynamiques?**
  - Performance, fiabilité, disponibilité, cohérence, …
- **…**

La Base de Connaissances



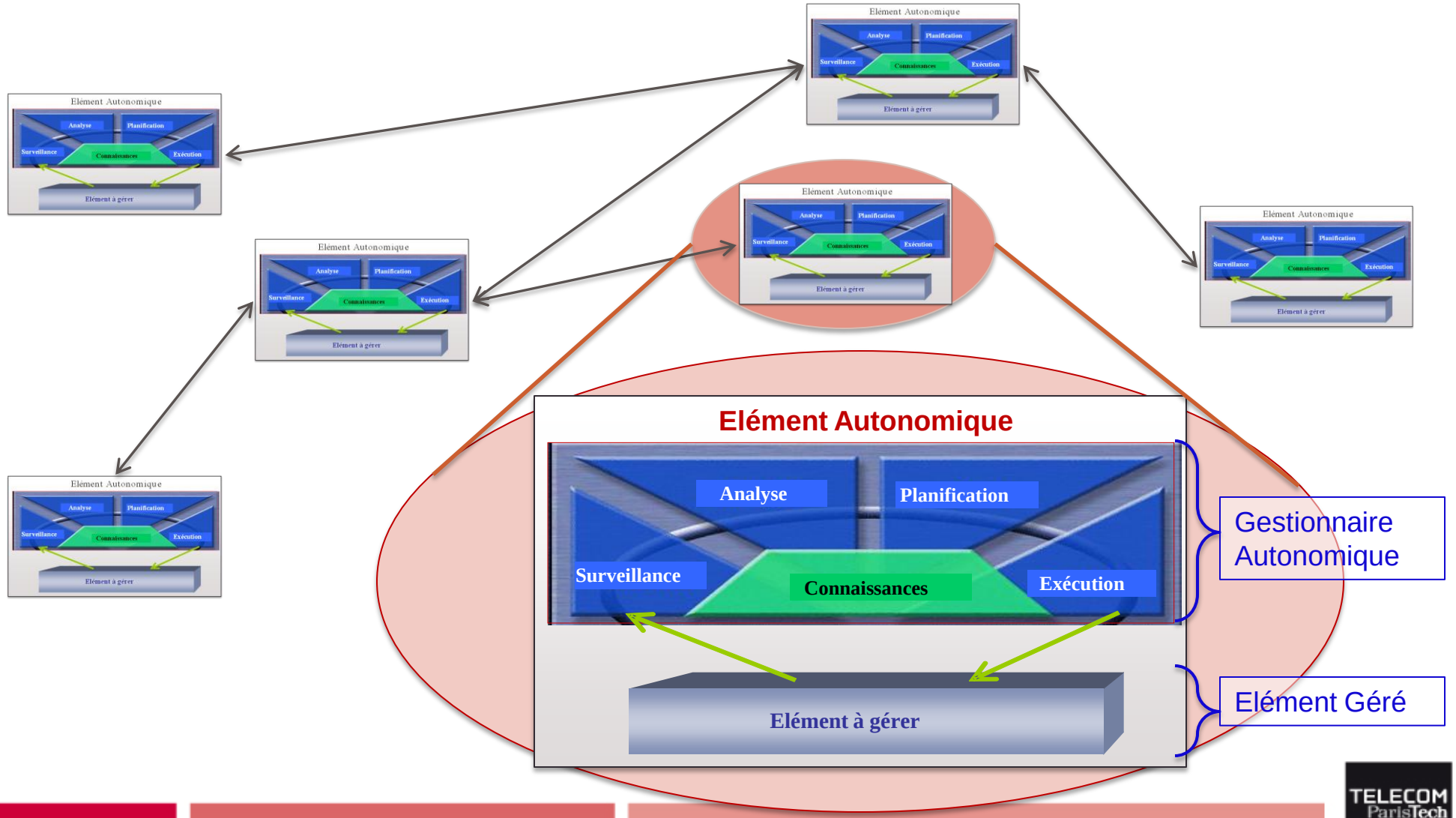
- **Représenter et faire évoluer**
  - Le système géré et son contexte d'exécution
    - Modélisation
    - Réflectivité (« reflection ») – la capacité d'un système de représenter sa structure et son comportement
  - Les buts d'autogestion
  - Le fonctionnement du système d'autogestion
    - Quels stratégies sont efficaces dans quels circonstances?

La Base de Connaissances - Problématiques



- **Comment obtenir et adapter les connaissances ?**
  - Spécification par les développeurs et les administrateurs système
  - Catégorisation, regroupement, « data mining », statistiques, …
- **Comment représenter les connaissances ?**
  - Clients hétérogènes
- **Comment gérer l'espace de stockage ?**
  - Combiens de temps garder une connaissance ?
  - Quelle longueur pour les historiques ?
- **Comment assurer l'accès aux connaissances ?**
  - Algorithmes de recherche ?
- **Comment passer à l'échelle ?**
  - Distribution? Décentralisation? Hiérarchisation?
- **…**

L'Elément Autonomique - Architecture



L'Elément Autonomique – Architecture (2)

- **L'élément autonomique – constituants :**
  - L'élément géré
    - Composant informatique « classique »
    - Offre éventuellement des points de surveillance et de contrôle
    - Ex : composant matériel (CPU, imprimante), ressource logiciel, application « legacy », système entier, …
  - Le gestionnaire autonomique
    - Surveille l'élément géré et son environnement externe
    - Analyse l'information collecté
    - Conçoit de plans pour résoudre des problèmes détectés
    - Exécute les solutions planifiées
    - => Facilite les tâches des administrateurs humains
- **Ceci est une architecture logique / conceptuelle - l'élément géré et le gestionnaire autonomique peuvent être séparés ou fusionnés**

Le Système Autonomique

- **Un système autonomique consiste de plusieurs éléments autonomiques interactives**
  - Eléments à plusieurs niveaux : ressource, service, application, économie mondiale, …
- **Chaque élément autonomique**
  - Offre des services aux autres éléments autonomiques et aux humains
  - Contient et utilise des ressources
  - Gère son comportement interne et ses relations avec d'autres éléments autonomiques, selon des politiques de haut niveau
- **Les politiques / objectives**
  - Les concepteurs les implantent dans les éléments
  - Des autres éléments avec plus d'autorité les imposent
  - Obtenues en signant des contrats avec des éléments de même niveau d'autorité
  - …
- **L'autogestion du système global résulte de l'autogestion de chaque élément ainsi que de l'interaction entre les éléments**

« The Vision of Autonomic Computing », J. O. Kephart, D. M. Chess, IEEE Computer, Jan. 2003
http://www.ece.rutgers.edu/~parashar/Classes/ece572-papers/06/kephart-ieeecomputer-03.pdf

L'Intégration

- **Comment obtenir une application autonome à partir d'éléments autonomes?**
- **Approche :**
  - Centralisé – modèle et adaptation globale de l'application
    - Auto-adaptation (approche « top-down »)
    - Comment passer à l'échelle?
  - Décentralisé – modèles et adaptations locaux à chaque élément
    - Auto-organisation (approche « bottom-up »)
    - Comment assurer un comportement émergeant conforme?
  - Hybride (hiérarchique) – modèles et adaptations aux plusieurs niveaux d'abstraction et/ou d'autorité
    - Les modèles et les adaptations à un certain niveau sont traduites dans des modèles et des adaptations aux niveaux plus bases

Exemples de Projets liés à l'Autogestion dans l'académie

- **AutoMate – Rutgers University**
  - L'autogestion des systèmes de type grille
  - Architecture et canevas génériques basés sur les systèmes multi-agents, les technologies P2P, …
- **Autonomia - Arizona University**
  - Approche holistique pour l'autogestion de ressources et de services répartis
  - Architecture et canevas génériques - intégration hiérarchique de composants autonomiques
  - http://www.ece.arizona.edu/~hpdc/projects/AUTONOMIA
- **ROC (Recovery Oriented Computing) - Berkeley/Stanford**
  - Techniques pour la dépendance des services Internet. Autoréparation basé sur des micro-redémarrages (« micro-resets »)
  - http://roc.cs.berkeley.edu
- **Rainbow – Carnegie Mellon University (CMU)**
  - Canevas pour l'autogestion de systèmes à composants
  - Guidée par les modèles architecturaux
- **ACT (Adaptive CORBA Template) - SENS Laboratory at Michigan State University**
  - Permet l'adaptation des applications CORBA par l'intégration dynamique (« weaving ») du code dans l'ORB
  - http://users.cis.fiu.edu/~sadjadi/Software/ACT
- **… !!!**

Exemples de Projets Européens liés à l'Autogestion

- **BioNets - Biologically-inspired Communications**
  - L'autogestion de systèmes pervasifs
  - www.bionets.eu
- **ANA: Autonomic Network Architecture**
  - Nouvelles techniques pour l'organisation et l'usage des réseaux
  - http://sourceforge.net/projects/ana
- **Shadows - A Self-healing Approach to Designing Complex Software Systems**
  - Technologies d'autoréparation pour l'amélioration de la fiabilité de systèmes complexes
- **Engineering Evolving Critical Systems – LERO**
  - http://www.lero.ie
- **…**

Exemples de Projets liés à l'Autogestion dans l'industrie

- **IBM Autonomic Computing toolkit**
  - Une collection de composants, outils, scenarios et documentations pour l'apprentissage et le développement de l'autogestion
  - http://www.ibm.com/developerworks/autonomic/overview.html
- **Intergiciels**
  - Jade / Jasmine - Serveur d'Application Java EE
    - Autoréparation par: la détection de pannes; le démarrage de serveurs; et le déploiement, l'installation et la configuration des logiciels
    - http://wiki.jasmine.ow2.org
  - La majorité de Serveurs d'Application (Java EE, .NET, …)
    - Auto-dimensionnement des caches et de pools d'instances et de connections
    - Reconfiguration de la distribution de charge
  - Joram – MOM
    - …
- **… !!!**

Support Technologique pour l'Autogestion
Quelques Exemples...

+ de flexibilité!

- **COS (Component-Oriented Software), SOA (Service Oriented Architecture)**
  - Facilite le changement à chaud (« hot-swap ») de parties de l'implantation de l'application
- **AOP (Aspect Oriented Programming)**
  - Facilite l'injection (dynamique) de fonctions dans l'implantation
- **JMX – Java Management Extentions**
  - Surveillance et modification des attributs d'objets Java (JavaBeans)
  - http://java.sun.com/javase/technologies/core/mntr-mgmt/javamanagement
- **iPOJO - technologie de services orientés composants, basée sur Java**
  - Autogestion d'interconnections entre les services; Surveillance et modification via JMX; Remplacement dynamique d'implantations des services; Notification de changements d'état de services (par événement)
  - www.ipojo.org
- **Patrons de conception (design patterns) facilitant le dynamisme**
  - Interception, chaîne de responsabilité, adaptateur, …
- **Systèmes P2P**
  - Permettent la persistance, la communication et la recherche de données sur des plates-formes distribuées fortement distribués, à large échelle et dynamiques
- **Systèmes de type Grille (« Grid Computing » et Nouage (« Cloud Computing»)**
  - Permettent l'approvisionnement dynamique de ressources (« dynamic resource provisioning ») - dans l'ordre de minutes (plutôt que de semaines)
- **… …**

Niveaux Applicatif et Intergiciel

Niveau Architectural

…

Problématiques principales de l'Autogestion « Grand Challenge » (1)

- **Niveau Conceptuel**
  - Identifier les abstractions et les modèles pour spécifier, comprendre, contrôler et implanter les comportements autonomiques
  - Adapter les théories existantes/classiques aux systèmes informatiques dynamiques
  - Offrir des modèles de négociation permettant aux gestionnaires autonomiques de former des relations
  - Concevoir des modèles statistiques permettant la création de modèles globaux de systèmes répartis et à large échelle, à partir de données locales (niveau service ou équipement)

Problématiques principales de l'Autogestion « Grand Challenge » (2)

- **Niveau Architectural**
  - Spécifier et construire des architectures permettant l'intégration des éléments autonomes dans des applications ou systèmes autonomes
  - Permettre la spécification d'objectifs d'autogestion au niveau local et global
  - Garantir la robustesse et la performance du système autonome résultant

Problématiques principales de l'Autogestion « Grand Challenge » (3)

- **Niveau Intergiciel**
  - Offrir des services pour l'implantation de fonctions d'autogestion fiables, efficaces, qui passent à l'échelle
  - Ex: surveillance de variables, découverte, communication par message, sécurité, identification, interconnexion dynamique, …

Problématiques principales de l'Autogestion « Grand Challenge » (4)

- **Niveau Applicatif**
  - Construire des applications et des systèmes capables de s'autogérer (auto-optimiser, réparer, configurer, protéger, ..)
  - Fournir des modèles de programmation, des canevas, des mécanismes de composition dynamique de composants, …
  - Fournir des outils qui facilitent la mise en œuvre, l'exécution et la gestion des applications autonomes

Problématiques de l'Autogestion et Quelques Objectifs du point de vue du Génie Logiciel

- **Les systèmes d'autogestions sont des systèmes complexes**
  - Les objectifs sont compliqués, conflictuels et en permanente évolution
  - Besoin de stratégies de gestion complexes et adaptables
  - Manque de support réutilisable pour faciliter le développement de solutions d'autogestion
- **Objectifs dans le domaine de l'ingénierie de logiciel**
  - Développer des canevas permettant la réutilisation de solutions partiels, existantes
  - Développer des infrastructures permettant l'intégration dynamique de composants / services
  - Développer des outils facilitant la spécification de l'autogestion d'un système par des non-experts
  - ….

Domaines de Recherche

- Intelligence Artificielle (conceptualisation, apprentissage automatique, rationnement, planification, ..)
- Modélisation (MDA, MDSE, réflectivité …)
- Robotique
- Automatique, cybernétique, théorie du contrôle, …
- Systèmes informatiques auto-adaptables, sensibles au contexte
- L'auto-organisation, l'émergence, les systèmes décentralisés
- …
- Autres que l'informatique : biologie, sociologie, économie, écosystèmes, …

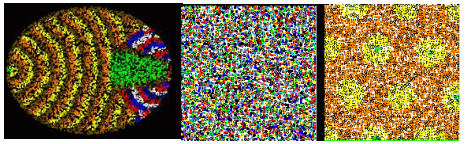
Autonomic Computing - Bibliographie

- **Sites web**
  - IBM Recherche - www.research.ibm.com/autonomic
  - IBM Développement - www-01.ibm.com/software/tivoli/autonomic
  - Autonomic Computing portal - www.autonomiccomputing.org
- **Articles de référence**
  - The Manifesto : « Autonomic Computing: IBM's Perspective on the State of Information Technology », Paul Horn, Oct. 2001 www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf
  - « The Vision of Autonomic Computing », Jeffrey O. Kephart, David M. Chess, IEEE Computer, Jan. 2003 - www.research.ibm.com/autonomic/research/papers/AC_Vision_Computer_Jan_2003.pdf
  - « The Dawning of the autonomic computing era », A. Ganek and T. Corbi, IBM Systems Journal, 42(1):5-18, 2003 - www.ece.rutgers.edu/~parashar/Classes/ece572-papers/06/ganek-ibmsysj-03.pdf
  - « Autonomic Computing: An Overview », M. Parashar, S. Hariri, Springer Verlag, Vol. 3566, pp. 247-259, 2005 - http://www.ece.rutgers.edu/~parashar/Classes/ece572-papers/06/parashar-upp-05.pdf
  - « The Autonomic Computing Paradigm », S. Hariri et al, Cluster Computing: The Journal of Networks, Software Tools, and Applications, Kluwer Academic Publishers, Vol. 8, No. 5, 2006 - www.ece.rutgers.edu/~parashar/Classes/ece572-papers/06/hariri-jcc-06.pdf
  - « An architectural blueprint for autonomic computing », IBM white paper, June 2006, www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf
- **Livres**
  - « Autonomic Computing: Concepts, Infrastructure, and Applications », Manish Parashar, Salim Hariri, CRC 2006
  - « Autonomic Computing », Richard Murch, IBM Press 2004

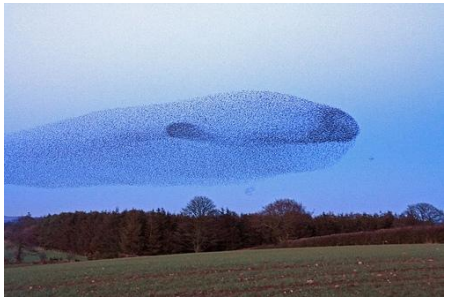
Initiatives similaires

- **Organic Computing**
  - http://www.sra.uni-hannover.de/orgcomp/home.html

- **Amorphous Computing**
  - http://groups.csail.mit.edu/mac/projects/amorphous



- **Biologically Inspierd Computing**
  - Swarm Intelligence
  - Evolutionary Computing



- **…**