Reinforcement Learning of Context Models for Ubiquitous Computing

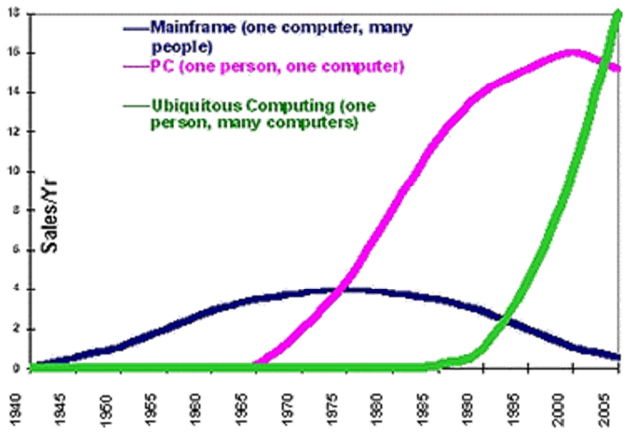
Sofia ZAIDENBERG
Laboratoire d'Informatique de Grenoble
PRIMA Group

Under the supervision of
Patrick REIGNIER and James L. CROWLEY

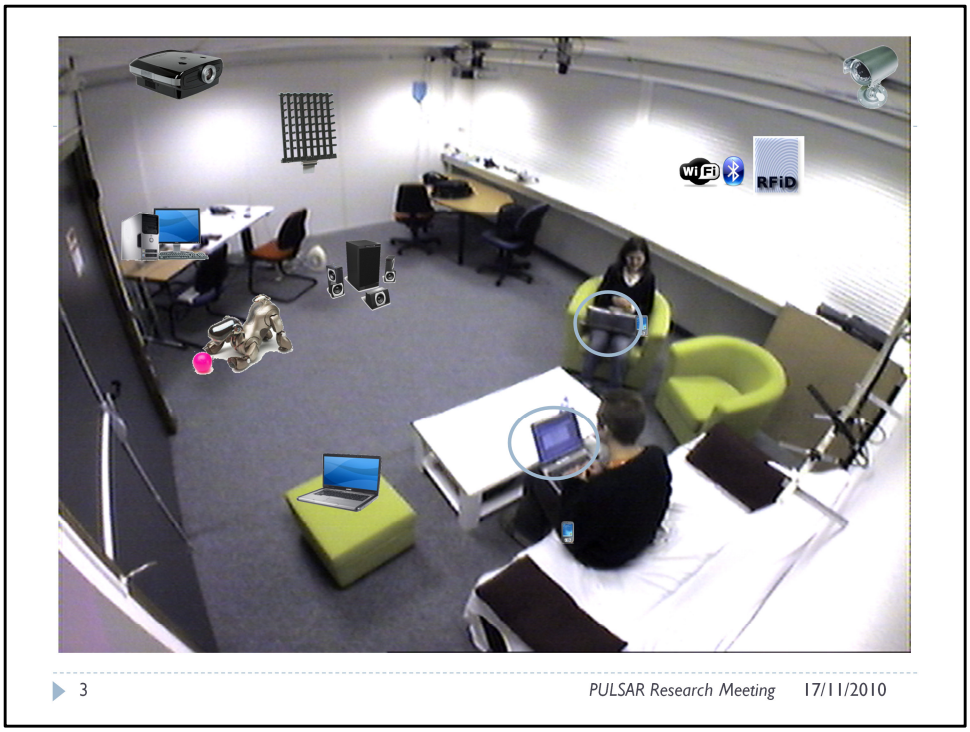
Ambient Computing

Ubiquitous Computing (ubicomp)

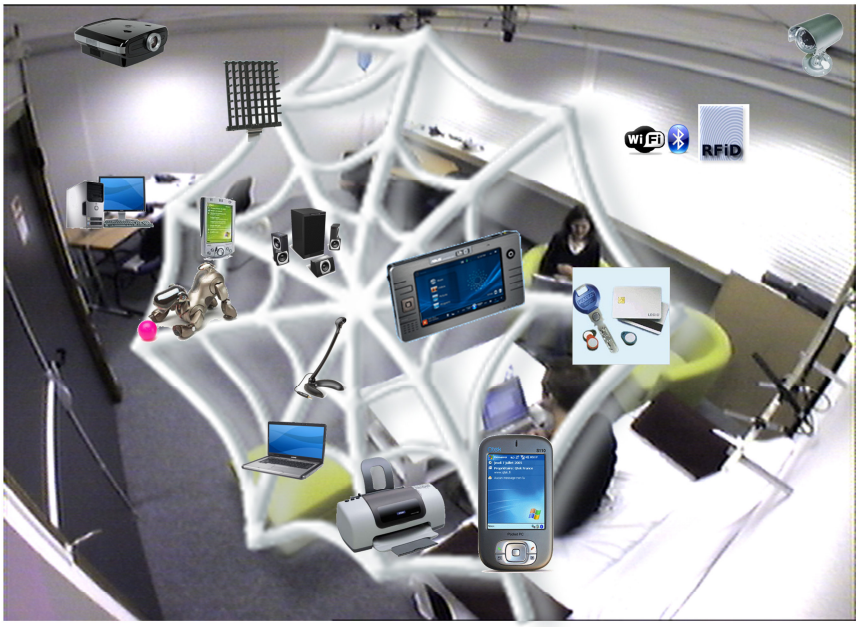
[Weiser, 1991]
[Weiser, 1994]
[Weiser and Brown, 1996]



Our research concerns the field of AmI, which is based on UbiComp.
UbiComp was introduced by Weiser in the early 90 and is based on the fact that we are surrounded by more and more devices.



Our everyday environments are filled with devices that can connect to each other (wifi, bluetooth, rfid…)



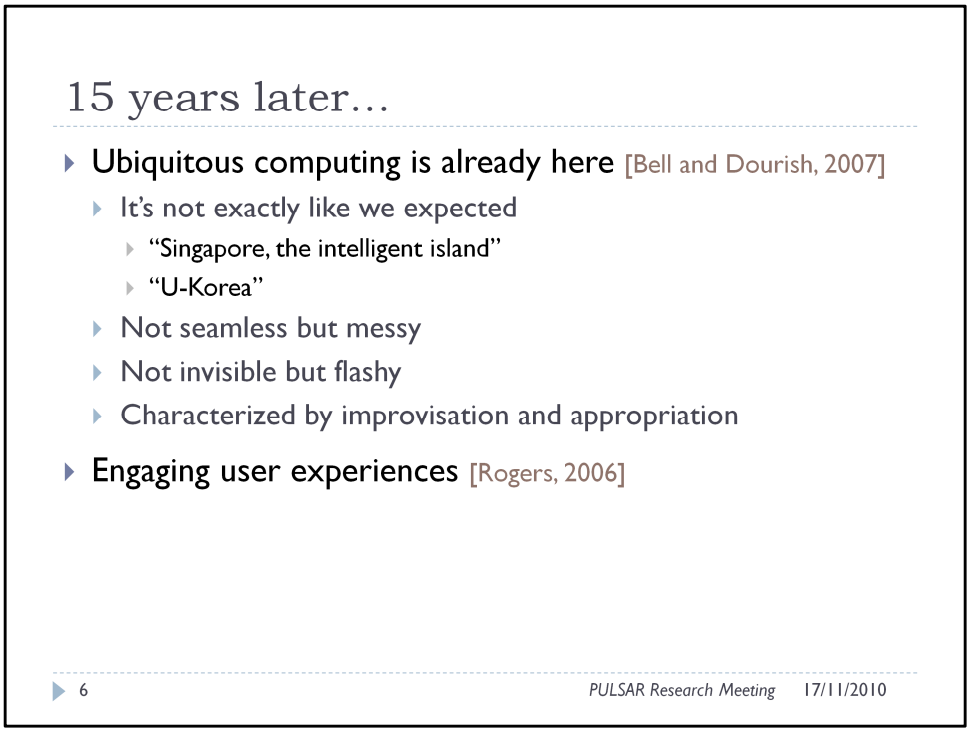
These devices are isolated because they were brought here by different people, at different times, they have nothing to do with each other.

We want to connect together these devices to create an undivided computing environment.
Together they can cooperate, synchronize and form a virtual computer, an intelligence spread out, distributed in all those atomic devices.

Ambient Computing

- *« autistic »* devices
  - Independent
  - Heterogeneous
  - Unaware
- Ubiquitous systems
  - Accompany without imposing
  - In periphery of the attention
  - *Invisible*
  - *Calm computing*

We want them to cooperate in order to create a Seamless, unobtrusive, invisible computing environment serving the user.



Acknowledgment: ubicomp still seems to be "just around the corner", why?

by technologies lashed together and maintained in synch only through considerable efforts; by surprising appropriations of technology for purposes never imagined by their inventors and often radically opposed to them;

"UbiComp technologies are designed not to do things for people but to engage them more actively in what they currently do."

New directions

- Study habits and ways of living of people and create technologies based on that (and not the opposite)
  [Barton and Pierce, 2006; Pascoe *et al.*, 2007; Taylor *et al.*, 2007; Jose, 2008]
- Redefine smart technologies [Rogers, 2006]
- Our goal:
  - Provide an AmI application assisting the user in his everyday activities

Visionary scenarios in UbiComp

Too much "magic" [Barton and Pierce, 2006] and oversimplification of the mundane nature of everyday life may render the scenario useless.

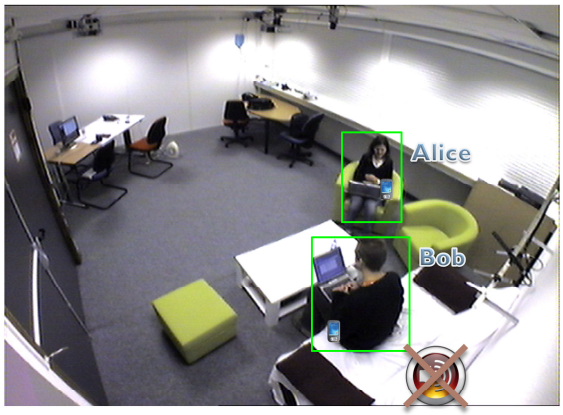
Consequences of inappropriate scenarios

Unrealistic systems that no one will adopt
Biased evaluations
Miss the real potential of a technology

Smart technologies are smart not because they were difficult to implement, but because their usage by people makes them smart

Our goal

- **Context-aware computing** + Personalization
  - Situation + user ⇨ action



1. Perception
2. Decision

8 *PULSAR Research Meeting* 17/11/2010

Intelligence:

capacity to adapt to its environment
perceive the environment and the situation
select and execute an action that
modifies the environment
or
modifies the agent itself
in order to achieve a more desirable state

The ubiquitous system should be as personal as a computer desktop (icons, fonts, colors…)

Intelligence dans un système ubiquitaire :

Percevoir la situation de l'utilisateur (le **contexte**)
Lui rendre un service adéquat

Exemple : Toujours mettre le téléphone de Bob en vibreur lorsque d'autres personnes sont présentes, toujours en sonnerie lorsqu'il est seul.

Proposed solution

Personalization by

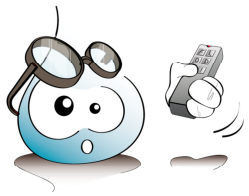
Learning

Outline

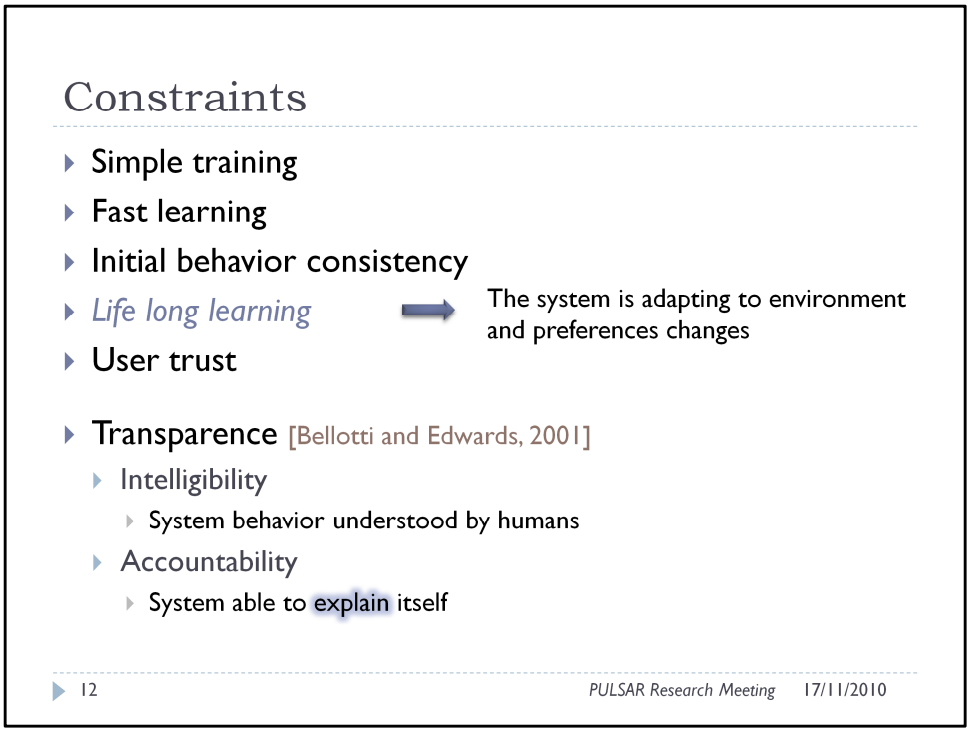
- Problem statement
- **Learning in ubiquitous systems**
- User Study
- Ubiquitous system
- Reinforcement learning of a context model
- Experimentations and results
- Conclusion

Proposed system

- A **virtual assistant** embodying the ubiquitous system
- The assistant
  - Perceives the context using its sensors
  - Executes actions using its actuators
  - Receives user feedback for training
  - Adapts its behavior to this feedback *(learning)*



Learn individual preferences during interactions between user and environment



Training non-intrusive, not a burden.

Trust: how can the user be comfortable with a system that does things instead of him?
If trust not gained, system is rejected

Learning allows a personalization with a minimum effort

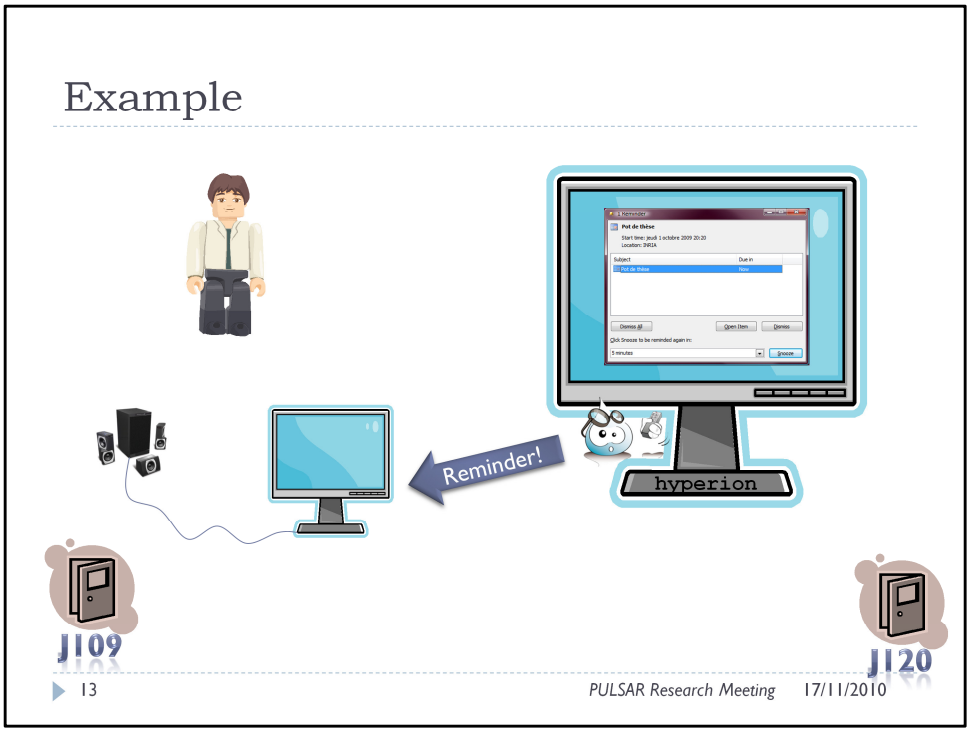
Transparence (not a black box):

Intelligibility: Context-aware systems that seek to act upon what they infer about the context must be able to represent to their users what they know, how they know it, and what they are doing about it.

Accountability: Context-aware systems must enforce user accountability when, based on their inferences about the social context, they seek to mediate user actions that impact others.

System evolves progressively

⇨ user has time to get used to the changes, gain trust



L'assistant personnel s'exécute sur la machine de bureau de l'utilisateur. Il détecte un rappel de l'agenda et décide de le transmettre à l'utilisateur. L'utilisateur se trouve dans le bureau J109, qui est équipé de haut-parleurs, l'assistant connait la machine à laquelle ils sont reliés, il peut donc lui envoyer le texte qui sera prononcé par synthèse vocale. L'utilisateur peut ensuite donner son avis sur ce service.

Outline

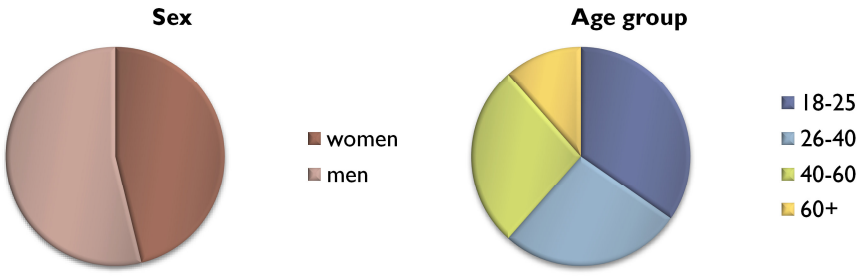
- Problem statement
- Learning in ubiquitous systems
- **User Study**
- Ubiquitous system
- Reinforcement learning of a context model
- Experimentations and results
- Conclusion

User study

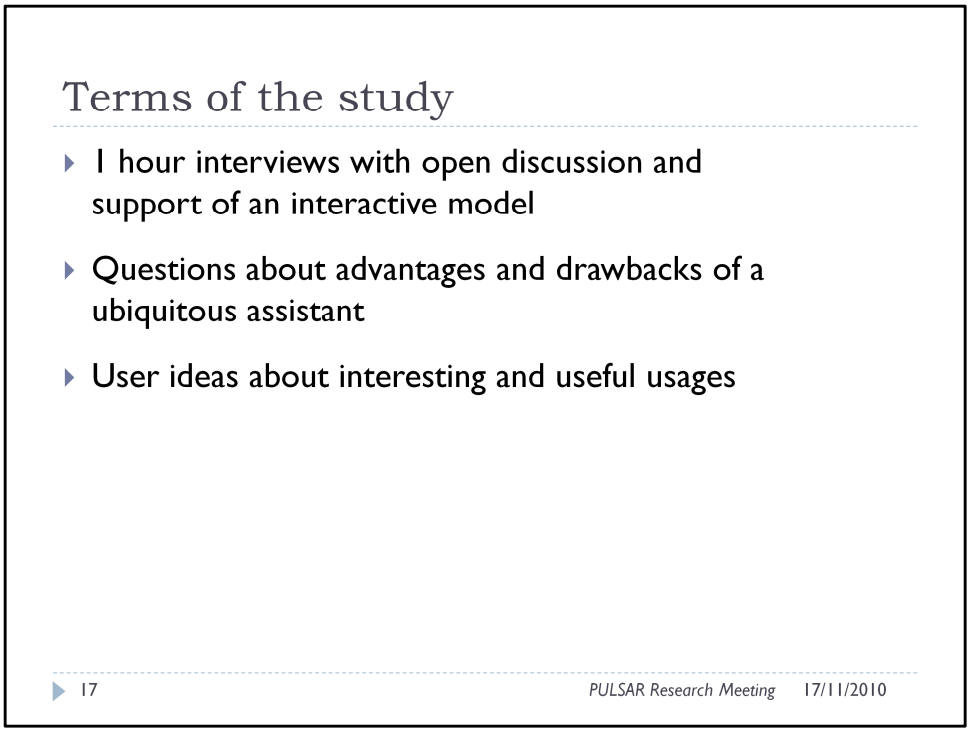
- Why a general public user study?
  - Original Weiser's vision of *calm computing* revealed itself to be unsuited to current user needs
- Objective
  - Evaluate the expectations and needs vis-à-vis "ambient computing" and its usages

Terms of the study

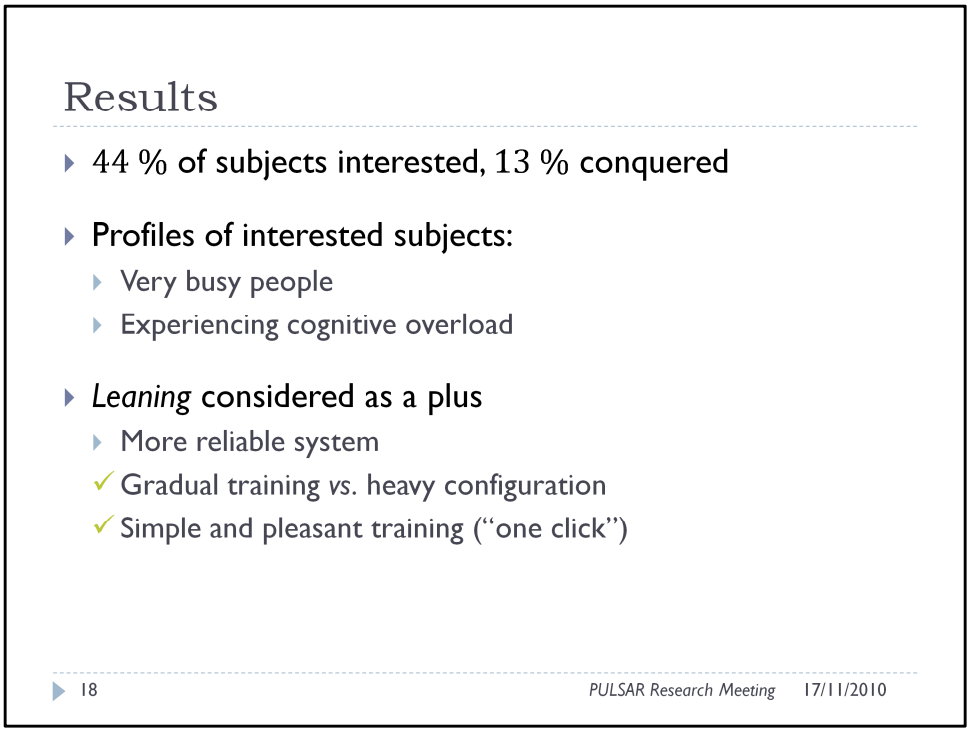
- 26 interviewed subjects
  - Non-experts
  - Distributed as follows:



Surjets non-informaticiens.



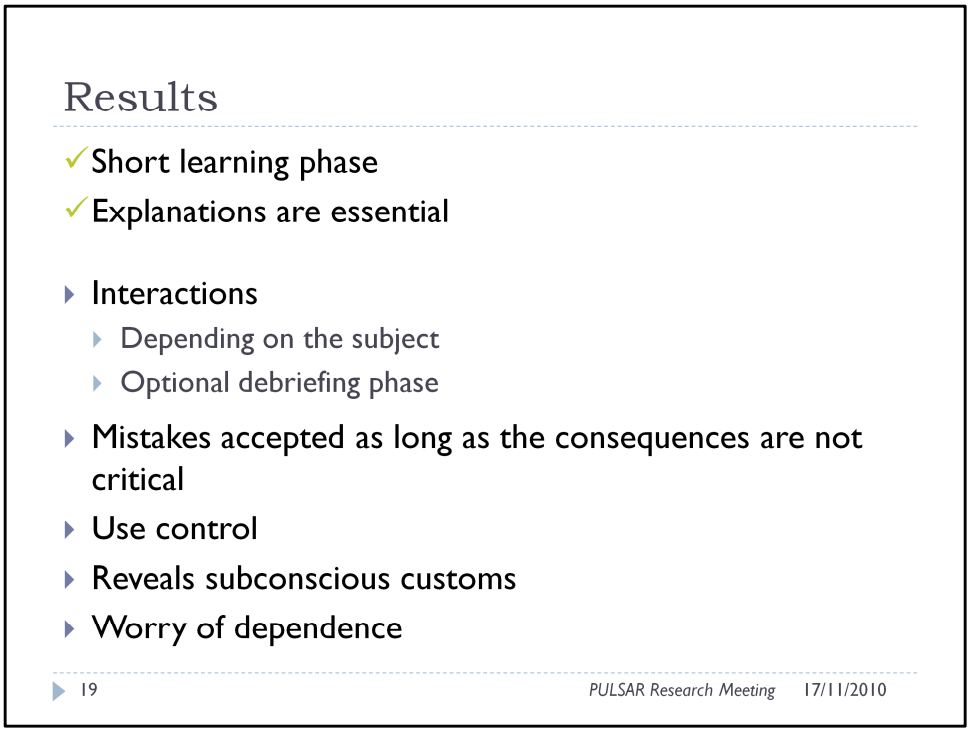
Surjets non-informaticiens.



Profils des sujets intéressés : personnes ayant un emploi du temps très chargé et dynamique, mêlant vie personnelle et professionnelle, personnes souhaitant une aide à l'organisation et à la gestion du temps.

Erreurs acceptées si l'utilisateur sait que le système apprend et si le système apporte un plus.

Cette enquête permet de justifier notre recherche, donc on a cherché à savoir si nos contraintes étaient bonnes et s'il y a d'autres éléments à prendre en compte.

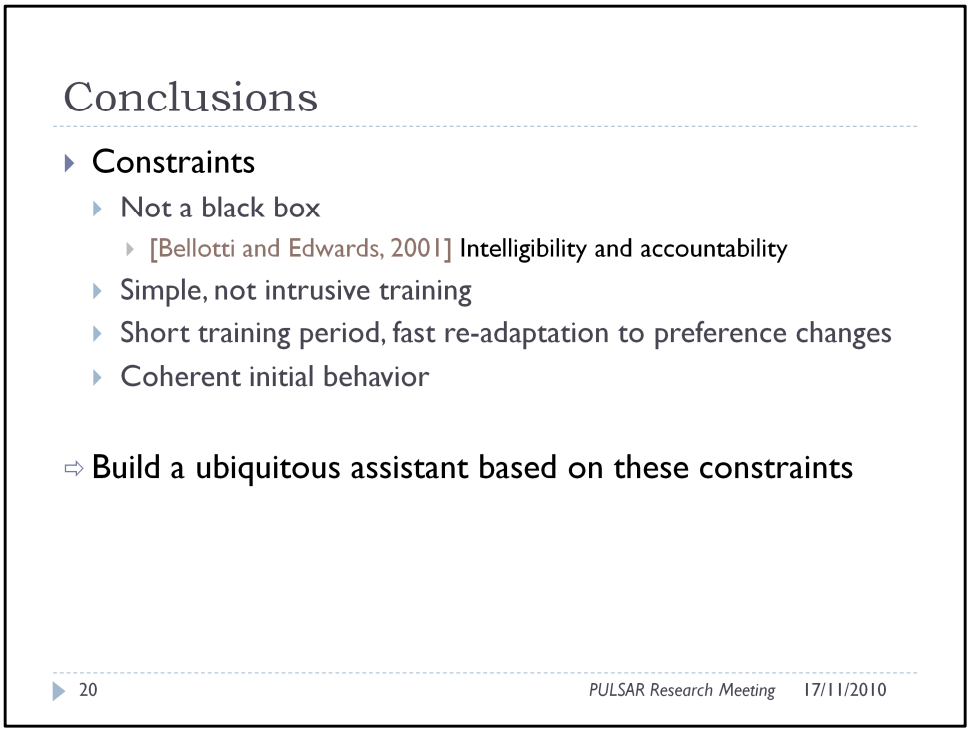


Initial learning phase short: 1 to 3 weeks

User has to keep control, have the last word, be able to shut the system down easily and immediately (red button).

The behavior that the assistant has learned reveals the user's subconscious habits.

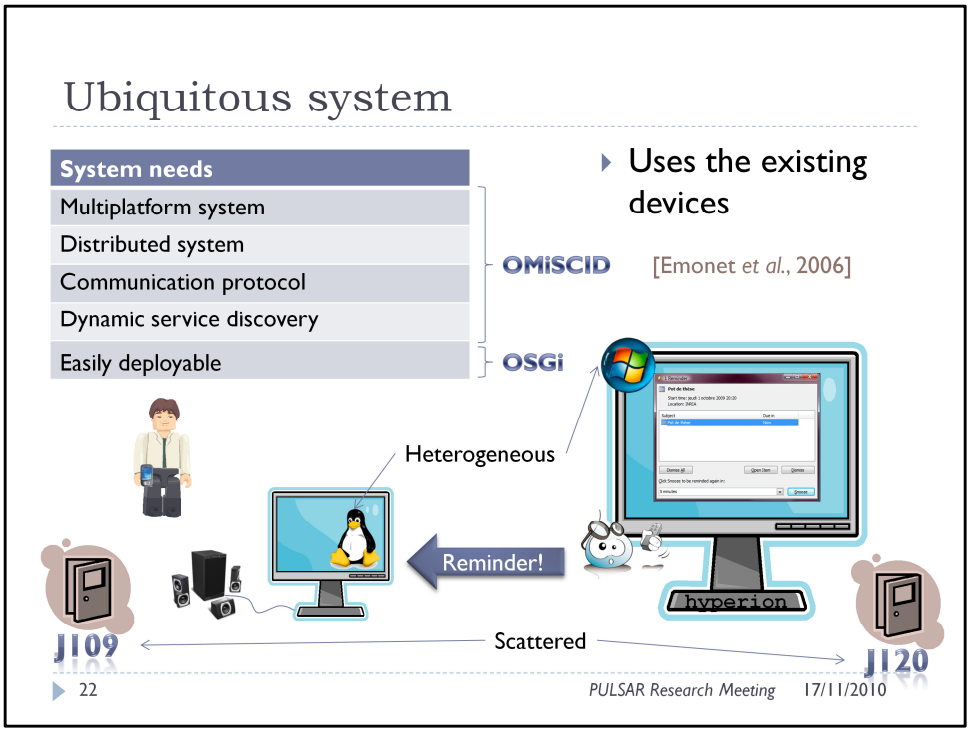
Worry to become dependent on a system that does things for us (what if the system is down?)



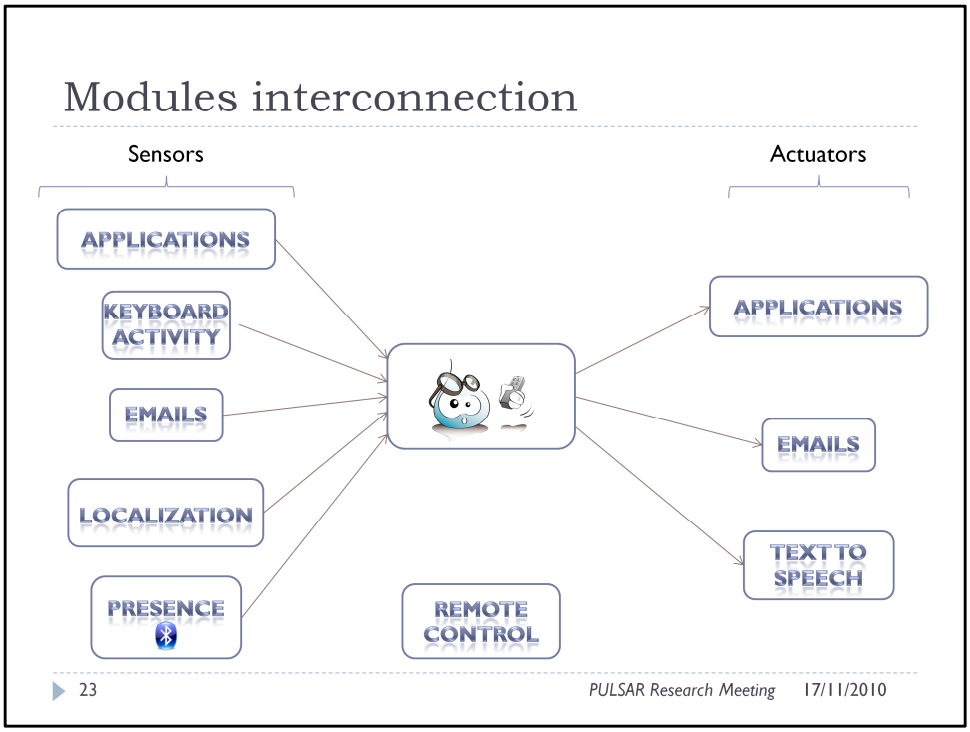
The system must not be a black box. As detailed in [3], a context-aware system can not pretend to understand all of the user's context, thus it must be responsible about its limitations. It must be able to explain to the user what it knows, how it knows it, and what it is doing about it. The user will trust the assistant (even if it fails) if he can understand its internal functioning.

Outline

- Problem statement
- Learning in ubiquitous systems
- User Study
- **Ubiquitous system**
- Reinforcement learning of a context model
- Experimentations and results
- Conclusion



Easy deployment to have an always working system (system always alive), should never be down for maintenance
→ Deployment on-the-spot
→ Distant administration
→ Handling of modules from a central repository → easy to update and add new functionalities without manual intervention



"Applications" actuator: manage software using standard systems of functionality export such as dcop on KDE

A module of our system is a bundle osgi and an omiscid service

Each device that is part of our system has an oscar platform with at least 2 running modules: remoteShell and omiscid, and can (opportunistic strategy) install dynamically and automatically any other module depending on the needs → obtain a flexible environment

remoteShell allows controlling the lifecycle of bundles in the same platform (installation, start, update)

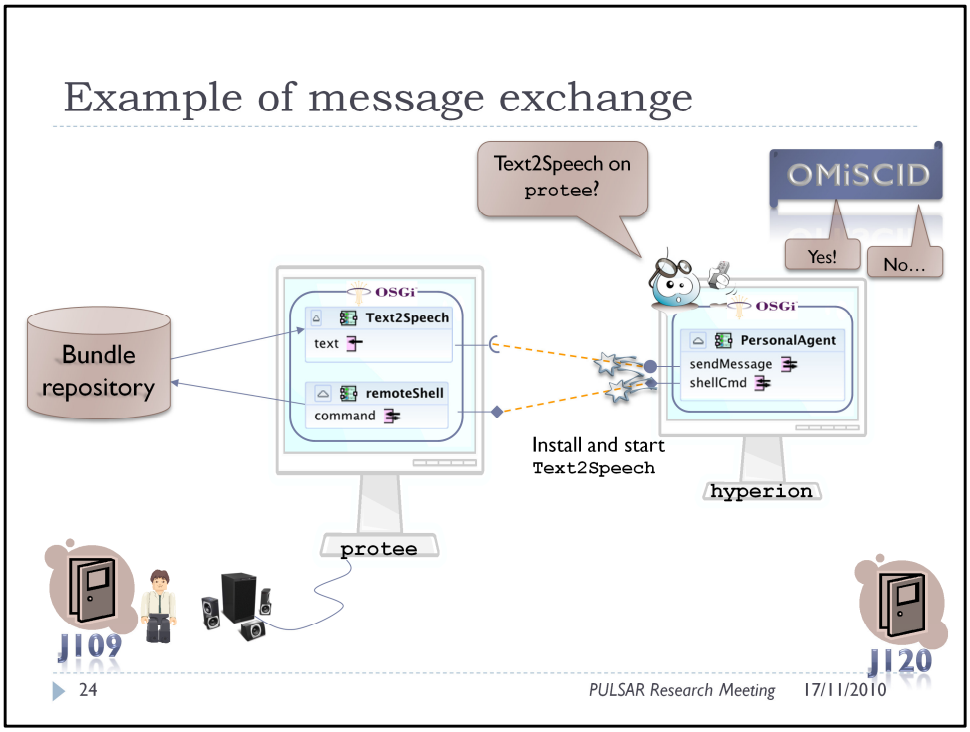
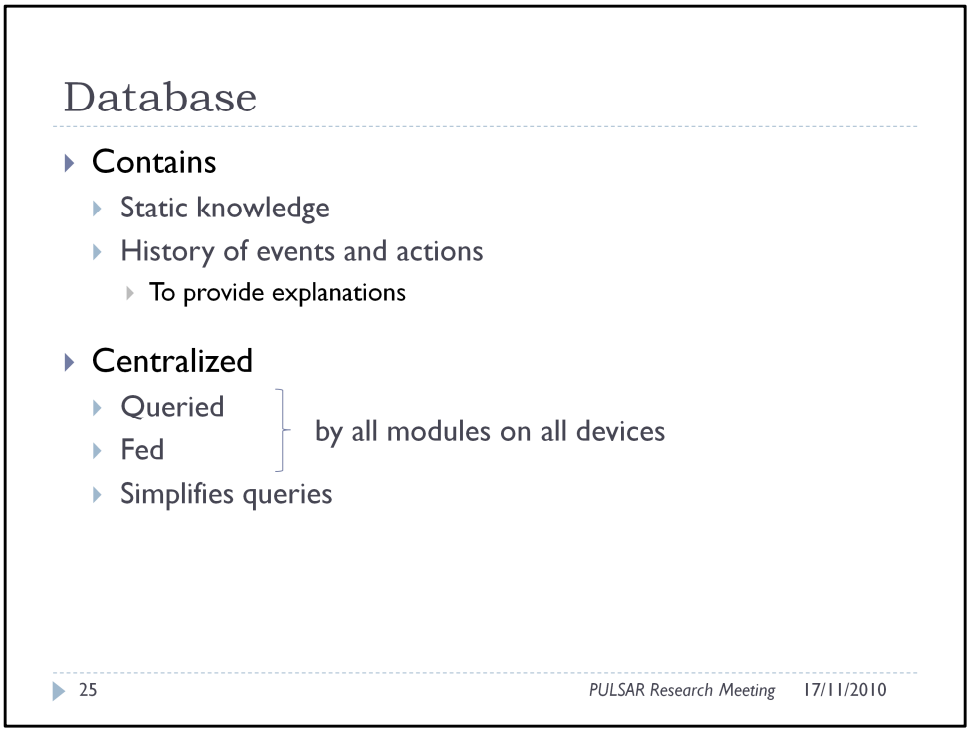


Illustration du déploiement à chaud (stratégie opportuniste) :
Exemple de transmission du rappel vu à un niveau plus bas (implémentation).

Le module AssistantPersonnel (« PersonalAgent ») s'exécute sur la machine de bureau de l'utilisateur. Il veut contacter le module « Text2Speech » (module de synthèse vocale) sur la machine « protee » (celle connectée aux haut-parleurs du bureau dans lequel se trouve l'utilisateur). C'est OMiSCID qui permet de brancher les modules les uns aux autres. Dans un 1er temps, OMiSCID répond à l'assistant que Text2Speech n'est pas en exécution sur protee. Alors l'assistant demande à OMiSCID d'obtenir une référence vers remoteShell (administration à distance) sur protee. On suppose que remoteShell doit toujours être disponible. L'assistant se branche sur remoteShell et lui envoie une commande pour installer et démarrer Text2Speech. remoteShell le fait à partir du dépôt central. Maintenant l'assistant redemande à OMiSCID une référence vers Text2Speech sur protee et l'obtient. Il s'y branche et lui envoie le texte à prononcer.

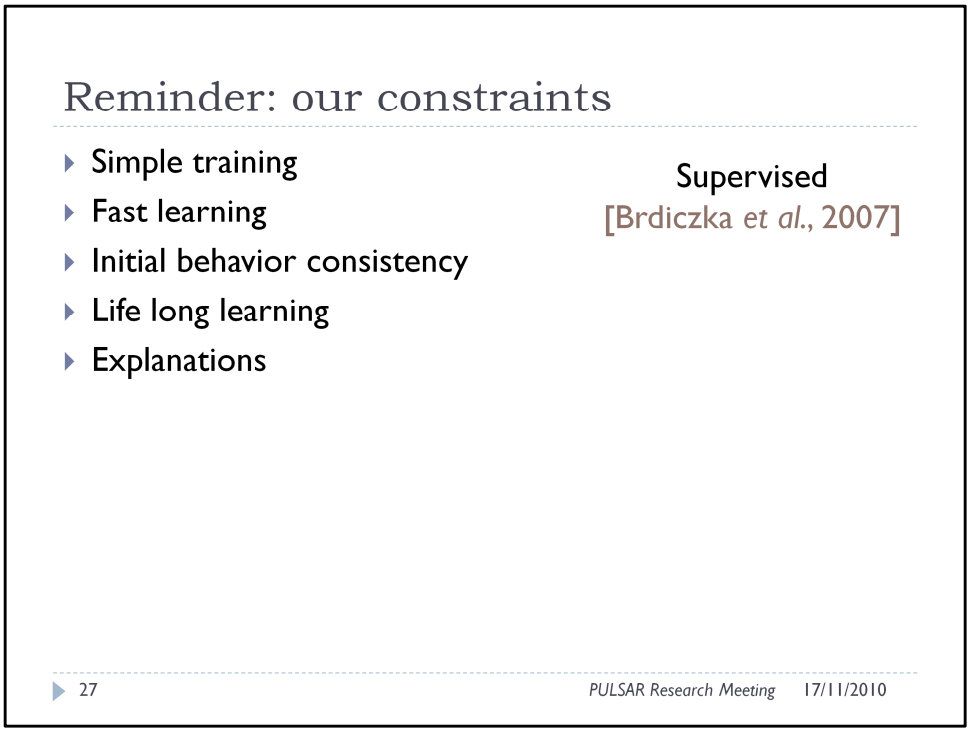


Connaissances statiques sur l'infrastructure et les utilisateurs (bureaux, adresses mail, dispositifs Bluetooth, etc.)

Historique des événements et actions du système et du cycle de vie des modules.

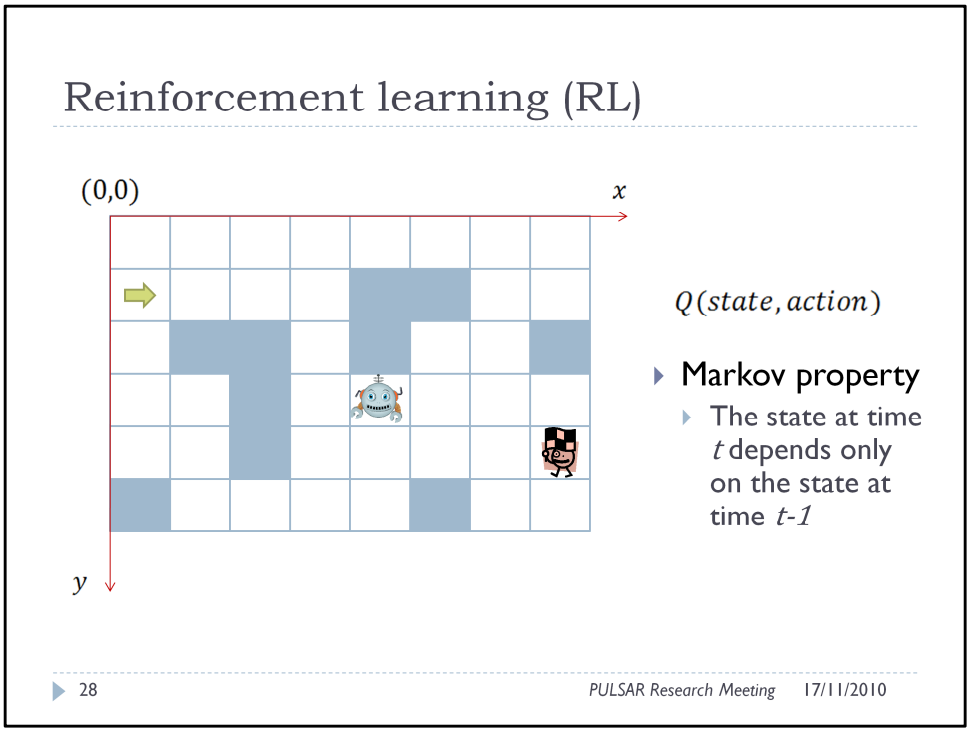
Outline

- Problem statement
- Learning in ubiquitous systems
- User Study
- Ubiquitous system
- **Reinforcement learning of a context model**
  - Reinforcement learning
  - Applying reinforcement learning
- Experimentations and results
- Conclusion



In supervised learning, the user had to label sequences afterwards, it's better if the user can give his feedback on-the-spot

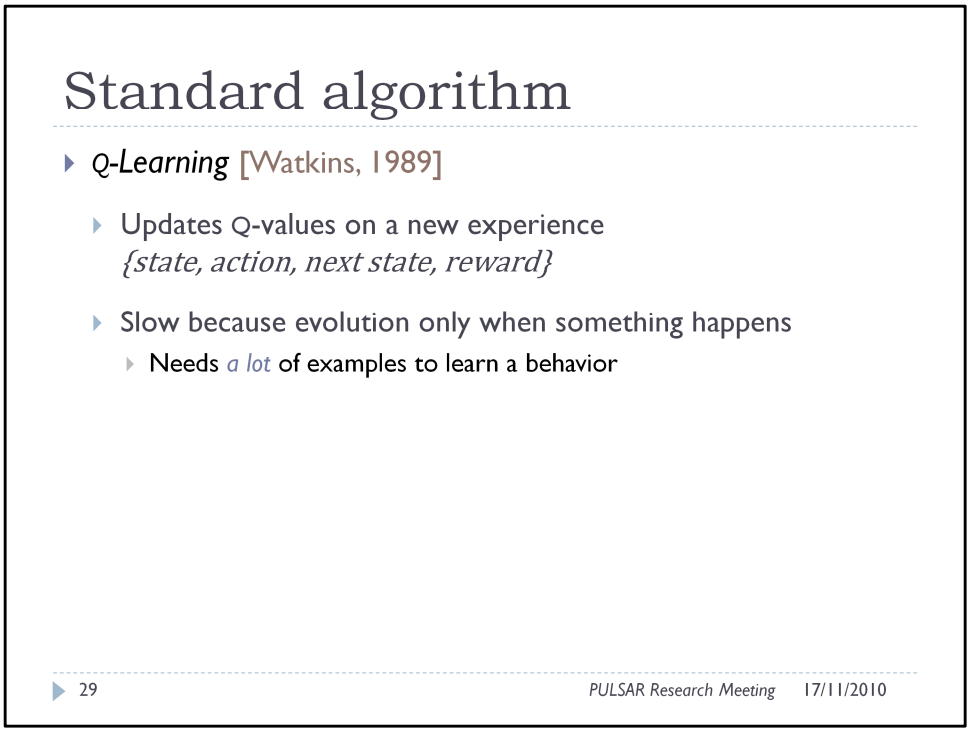
L'apprentissage par renforcement est adapté (ou peut l'être avec les modifications que nous allons apporter) à ces contraintes.



L'environnement est le labyrinthe, le robot connait son état dans l'environnement : (x, y), et peut faire des actions (se déplacer d'une case), reçoit des renforcements (but = très bien, dans le mur = très mauvais), doit maximiser la somme des renforcements reçus dans le temps.

Il explore l'environnement pour apprendre toutes les valeurs de qualité des couples état-action.

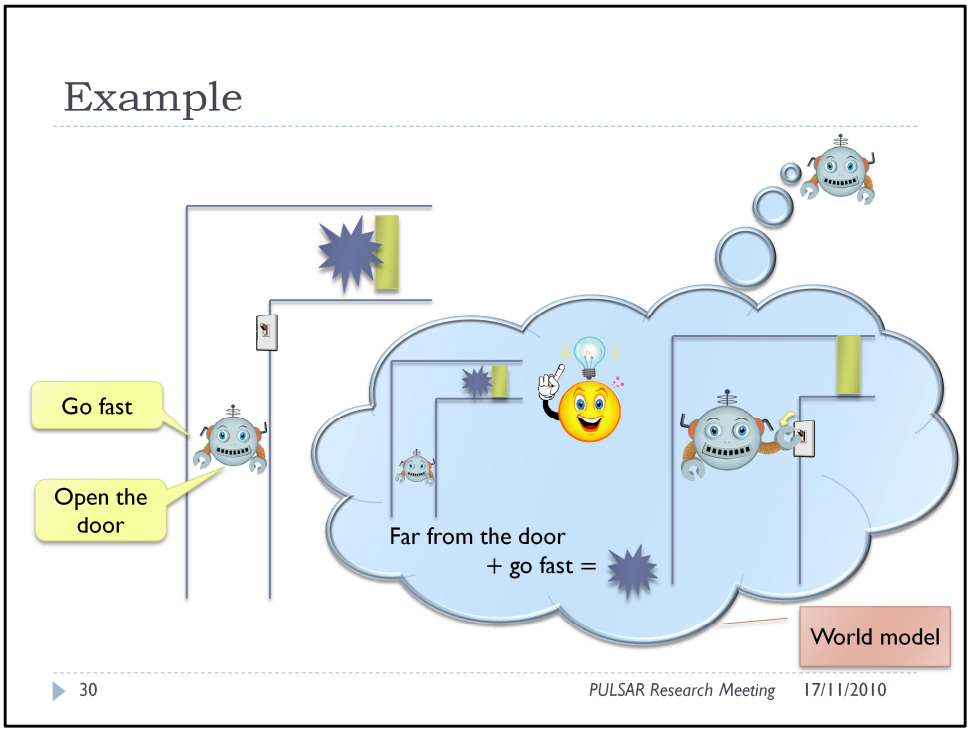
L'AR (apprentissage par renforcement) est bien adapté à des environnements contraintes, entièrement connus et maitrisés (potentiellement stochastiques). La difficulté de ce travail consiste à l'adapter à toutes les contraintes introduites par un environnement réel, complexe, et le fait que l'utilisateur est directement impliqué dans les actions de l'assistant.



User in the loop

Fast adaptation while respecting acceptability

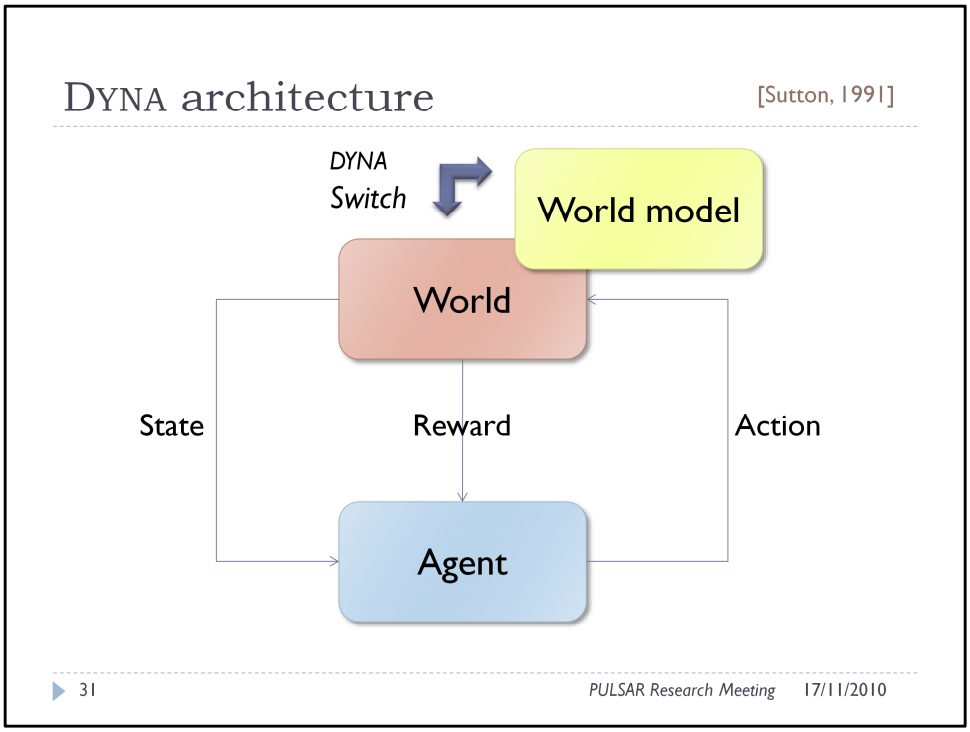
To make one learning step we wait for a user action → need to wait a long time and to see a lot of actions to learn a behavior.
We wish to do this process offline, but we don't know s' and r → we learn the world model from real interactions and we use it to learn Q-values.



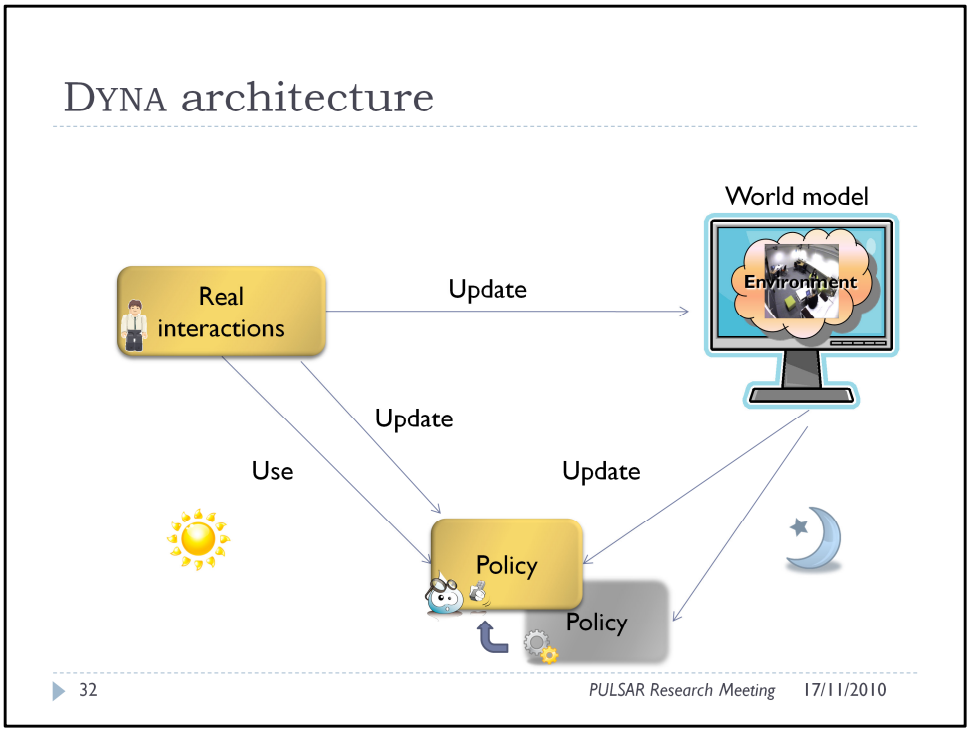
On voudrait revivre les expériences virtuellement, au lieu de les vivre dans le monde réel.
→Pour ça, on a besoin d'un modèle du monde réel.

On veut tirer le maximum de profit de chaque expérience.

Processus intrinsèquement exploratoire. Le système ne connait pas le monde et il va tâtonner. On ne raisonne pas. Le modèle permet de faire une phase de raisonnement hors ligne.
L'exploration est faite dans le modèle. Revivre l'expérience pour comprendre que c'est pas bien d'aller vite, et puis aller plus loin (explorer). Essayer de l'arrêter de plus en plus loin de la porte, et voir que ça ne suffit pas à pouvoir s'arrêter et éviter le choc, donc apprendre que même si on est loin de la porte, aller vite est dangereux.
Exploration indépendante des expériences vécues : faire l'action « activer interrupteur » dans le modèle et avoir une estimation des conséquences dès la 1ère fois qu'on est dans la même situation dans le monde réel.



Faire une partie de l'exploration dans le modèle, sans impliquer le monde réel.

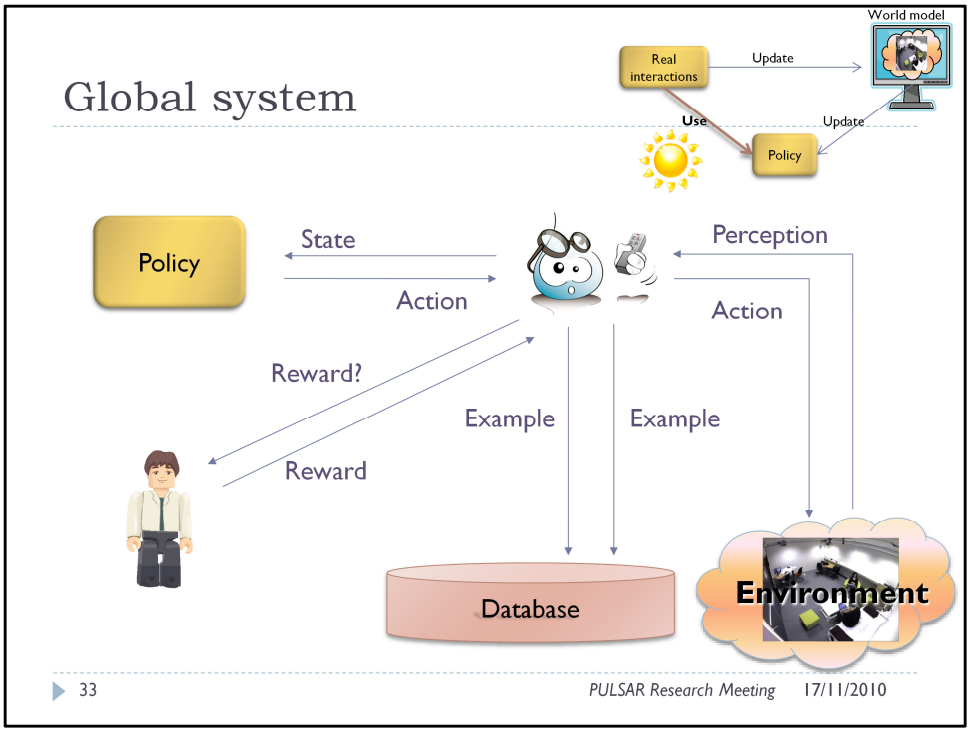


L'utilisateur qui interagit avec un système informatique (en général) construit inconsciemment un modèle mental de ce système.
Si le comportement du système change tout le temps, l'utilisateur n'arrivera pas à construire son modèle.
Un système dont on n'arrive pas à prédire les actions, qui surprend l'utilisateur, nuit à la confiance.

Donc on peut faire une 2ème politique interne (invisible à l'utilisateur), qui sera apprise en utilisant le modèle, sans impliquer l'utilisateur. La politique utilisée lors des interactions n'est pas modifiée sans prévenir l'utilisateur. On substitue la politique interne (nouvellement apprise) à la politique vue par l'utilisateur à certains moments bien définis (option de l'assistant : substitution tous les jours / toutes les semaines / à la demande, etc.).

Jour = fonctionnement interactif

Nuit = fonctionnement non interactif

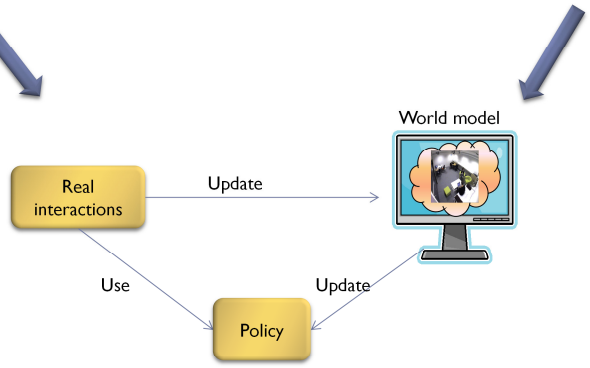


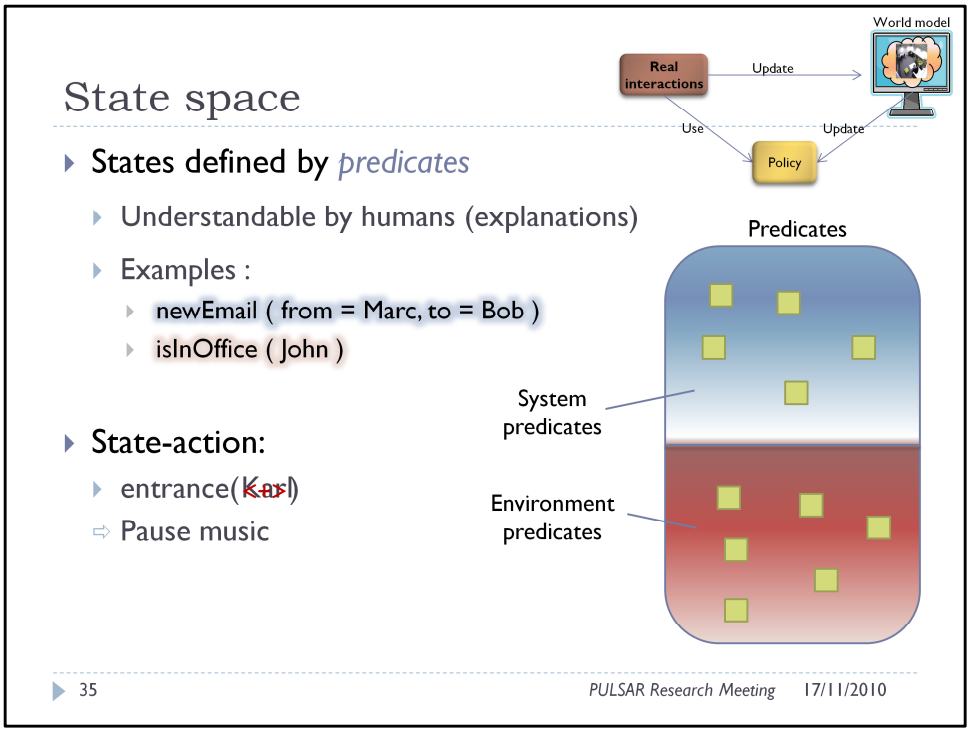
Exemples serviront à construire modèle du monde.

Partie interactive, sans apprentissage

Modeling of the problem

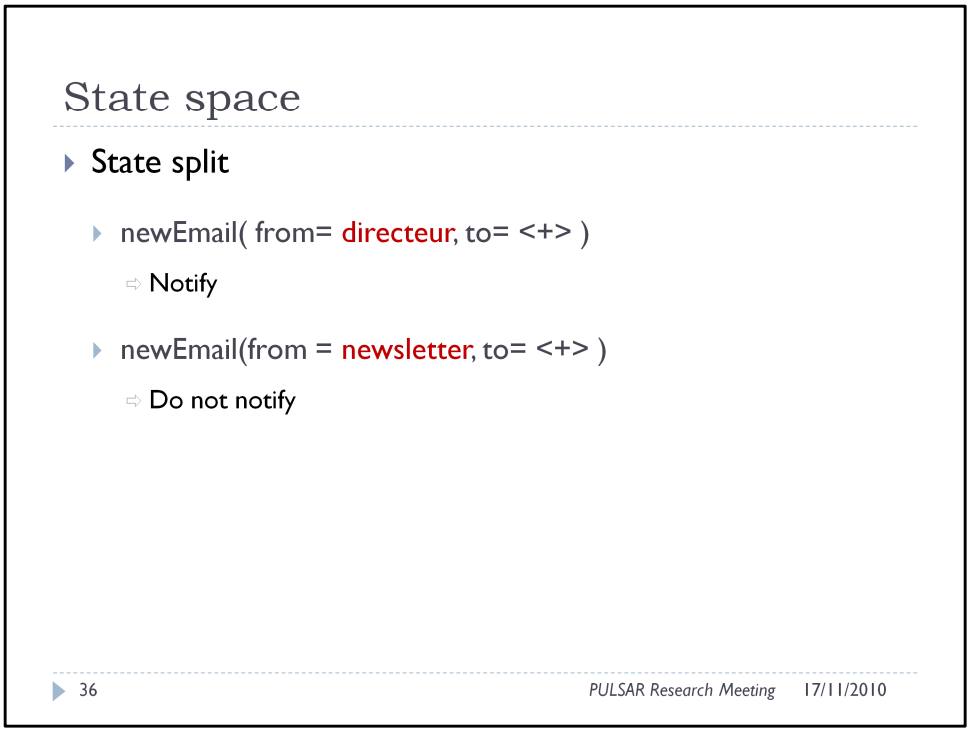
- Components:
  - States
  - Actions
- Components:
  - Transition model
  - Reward model



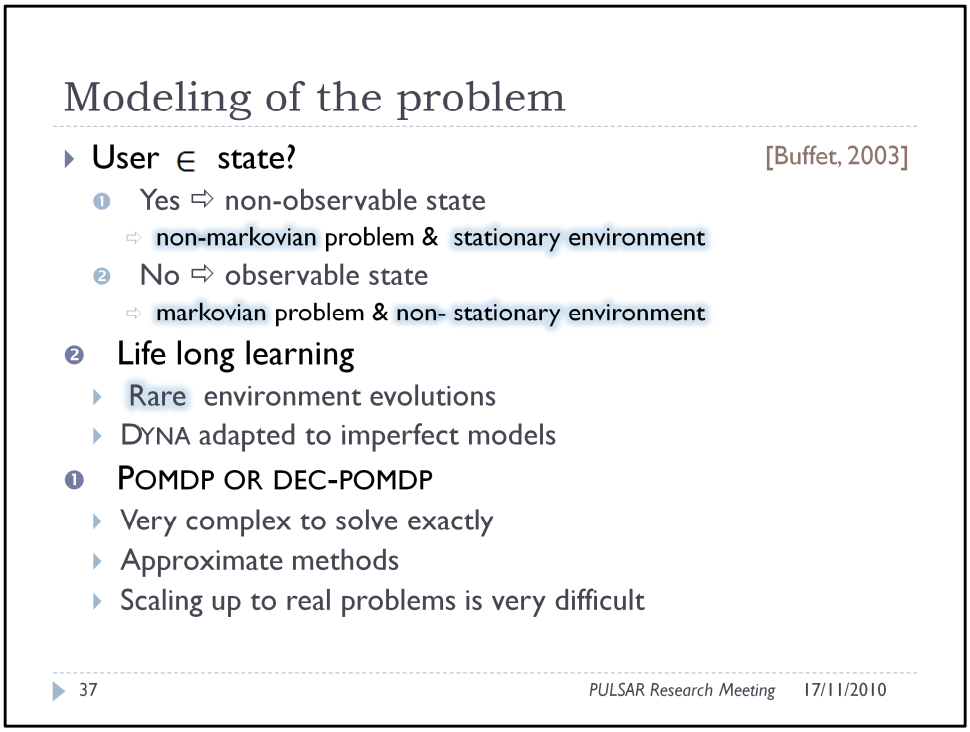


Prédicats du 1er ordre.

We cannot be too specific because there are too many states.
Factorizing (generalizing) early allow to get quickly a behavior, without that, the learning could not converge.
Make most of each experience.



Dans les cas où la factorisation n'est pas pertinente (nuit à la satisfaction de l'utilisateur vis-à-vis du service), on fait une division des états factorisés a postériori (post-traitement).



En AR classique, l'état de l'agent est modifié seulement par actions de l'agent.
Chez nous, l'état est modifié par les actions mais aussi pas des événements extérieurs ou l'utilisateur (réception d'un mail, etc.).
On a défini l'état avec des éléments qu'on peut observer, est-ce qu'en plus on y incorpore l'utilisateur ?

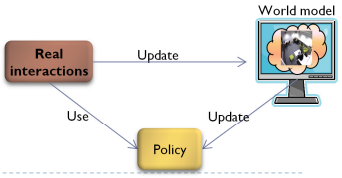
La non stationnarité provient des changements des préférences utilisateur.
Non stationnarité = conséquences des actions changent dans le temps → conséquence : je peux plus me comporter de la même façon, donc une action qui était bonne avant, ne l'est plus (les renforcements changent).
Ex : l'utilisateur aime bien lire son mail le matin, donc ouvrir son client mail le matin était une bonne action, puis il a changé ses habitudes et lit son mail l'après-midi, donc la même action (ouvrir client mail le matin) a une mauvaise conséquence.
Évolution monde provoquée par changement habitudes utilisateur.

Question : peut-on appliquer un apprentissage classique (comme dans le labyrinthe) dans un environnement non-stationnaire (imaginons que les obstacles du labyrinthe changent dans le temps…) ?
La réponse est oui, on peut car on suppose que utilisateur évolue lentement. L'environnement n'est pas stationnaire, mais il est quasi-stationnaire, il est localement stationnaire. Il est stationnaire pendant suffisamment longtemps pour qu'on puisse apprendre le bon comportement. Lorsqu'il change, on va le suivre. Par conséquent, on peut se ramener à un PDM.

On veut pas rentrer dans la problématique des PDMPO car c'est un problème trop complexe, on préfère rester dans le cadre mieux maitrisé des PDM.

Action space

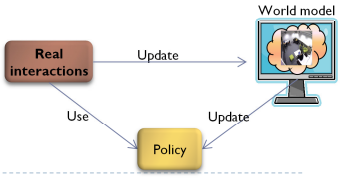
- Possible actions combine
  - Forwarding a reminder to the user
  - Notify of a new email
  - Lock a computer screen
  - Unlock a computer screen
  - Pause the music playing on a computer
  - Un-pause the music playing on a computer
  - Do nothing



Reward

- Explicit reward
  - Through a non-intrusive user interface

- Problems with user rewards
  - Implicit reward
    - Gathered from clues
      (numerical value of lower amplitude)
  - Smoothing of the model



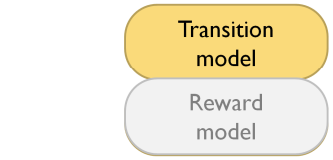
Récompenses données par l'utilisateur, ça pose différents problèmes (abordés dans le manuscrit).

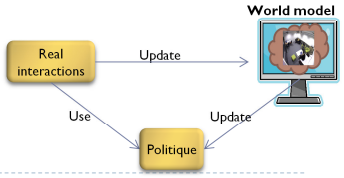
Maximiser l'utilité de chaque récompense reçue.

Use of eligibility traces

World model

- Built using supervised learning
  - From real examples
- Initialized using common sense
  - Functional system from the beginning
  - Initial model *vs.* initial Q-values [Kaelbling, 2004]
  - Extensibility



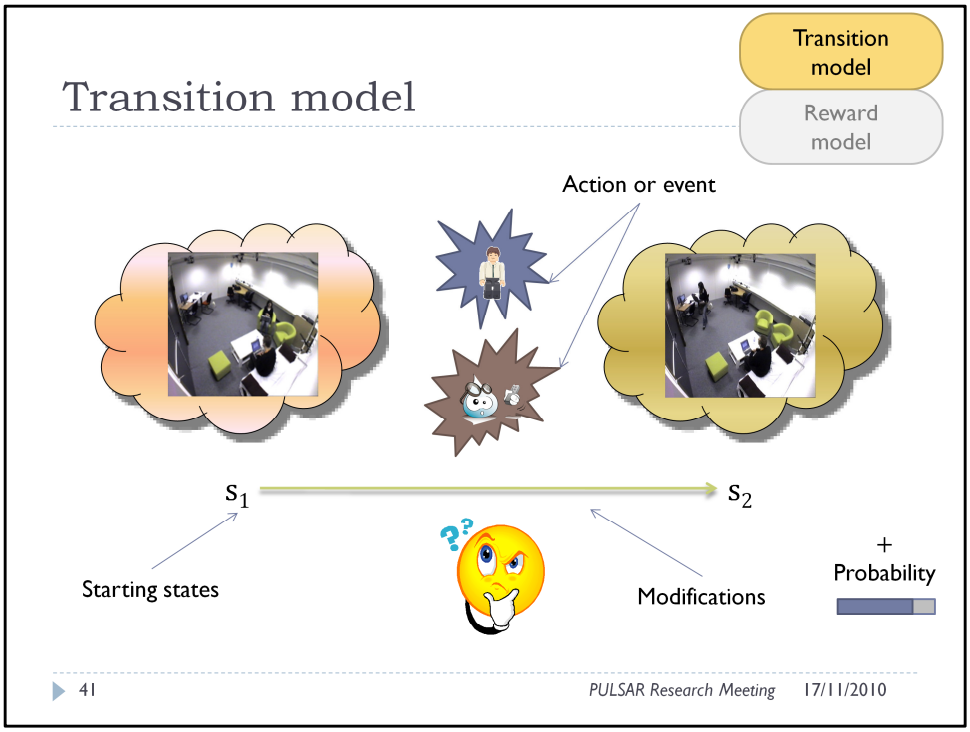


On collecte en permanence les données réelles pour améliorer et affiner le modèle.

Au premier lancement : on exécute AR en tâche de fond pour initialiser un comportement par défaut à partir des modèles par défaut.

On donne des modèles initiaux et pas des valeurs de qualité initiales car il est plus facile de spécifier un renforcement qu'un comportement [Kaelbling, 2004].

Modèle initial pourrait être utilisé pour initialiser un autre système d'apprentissage que l'AR, alors que la Q-table est propre à l'AR.



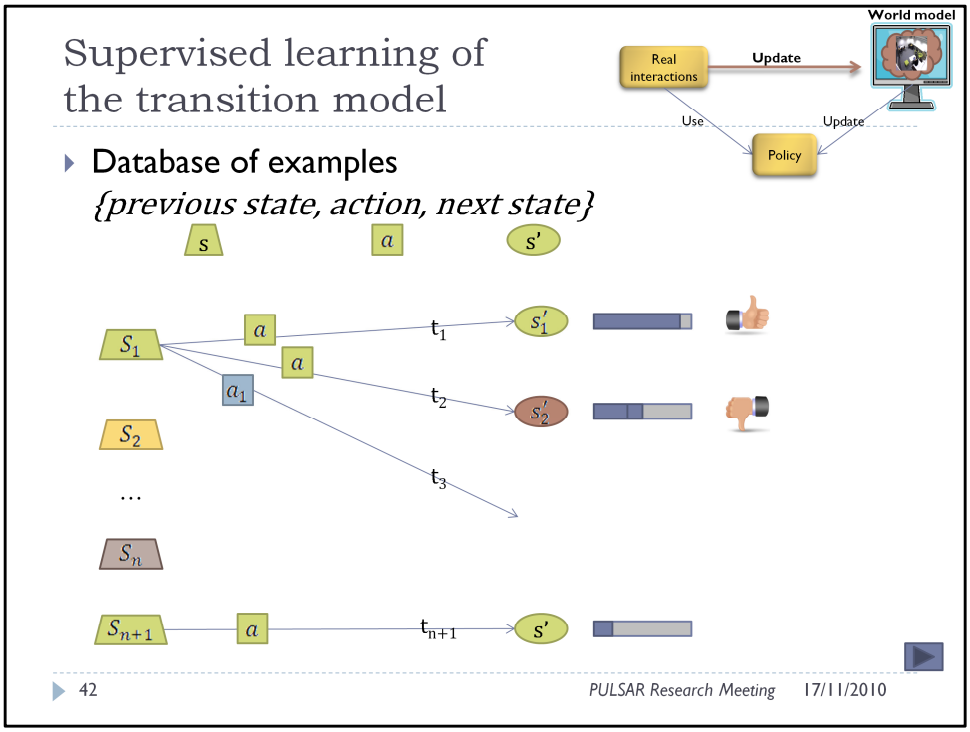
On veut apprendre ça mais pas trop précisément, en généralisant.
On veut comprendre comment ça se fait, le modéliser, on va calculer des transformations. Chaque transformation opère sur un état de départ factorisé.

Généralisation pour tirer le max de profit de chaque expérience, et pour permettre exploration dans le modèle plus tard.

Observation du monde n'est pas parfaite donc une même action peut donner des s2 différents --> probabilités (pour tenir compte de l'incertitude).

Modèle de transition = ensemble de transformations d'un état (factorisé) vers le suivant étant donnée une action ou un événement.
Une transformation est composée de

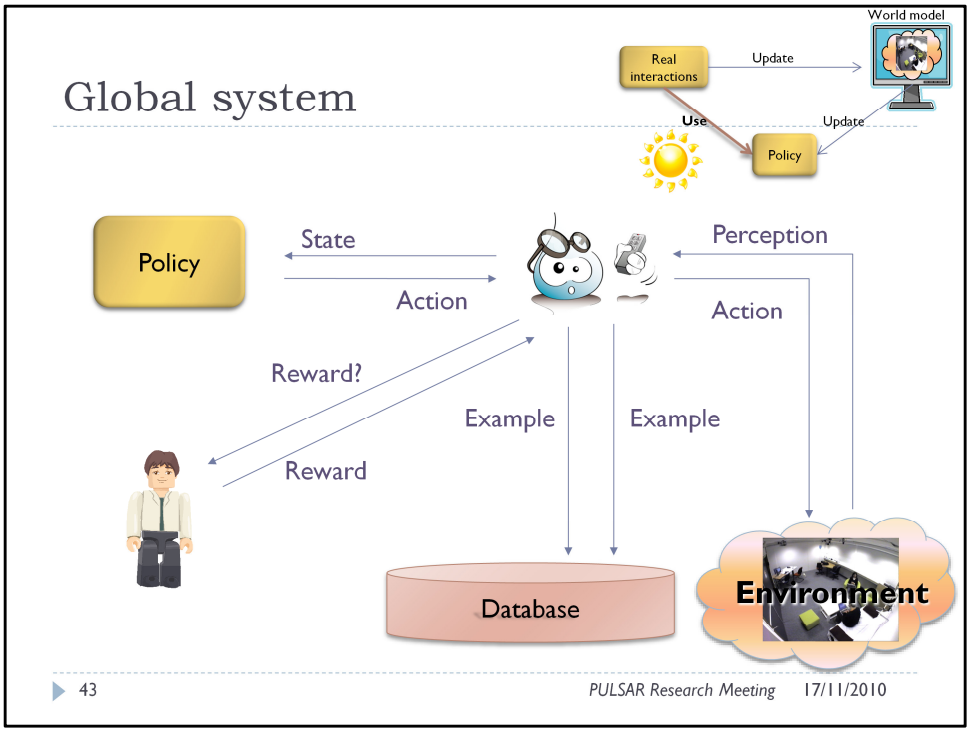
- Un état précédent
- Des modifications
- L'action (ou l'événement)
- Une probabilité



L'exemple renforce t1 et affaiblit t2, on cherche pas à stabiliser ces distributions car l'environnement est non stationnaire donc on sait qu'il va évoluer, et les probabilités doivent pouvoir évoluer avec lui. On ne diminue pas le poids des nouveaux exemples.

Nouvel exemple ⇨ nouvelle transformation générique pour faire de la généralisation.

Apprentissage « stable » : pour qu'un changement ait lieu dans le modèle, il faut voir l'exemple plusieurs fois (éviter d'apprendre les erreurs de capteurs). Si c'est un vrai changement, on aura forcément plusieurs exemples.

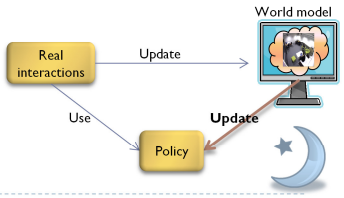


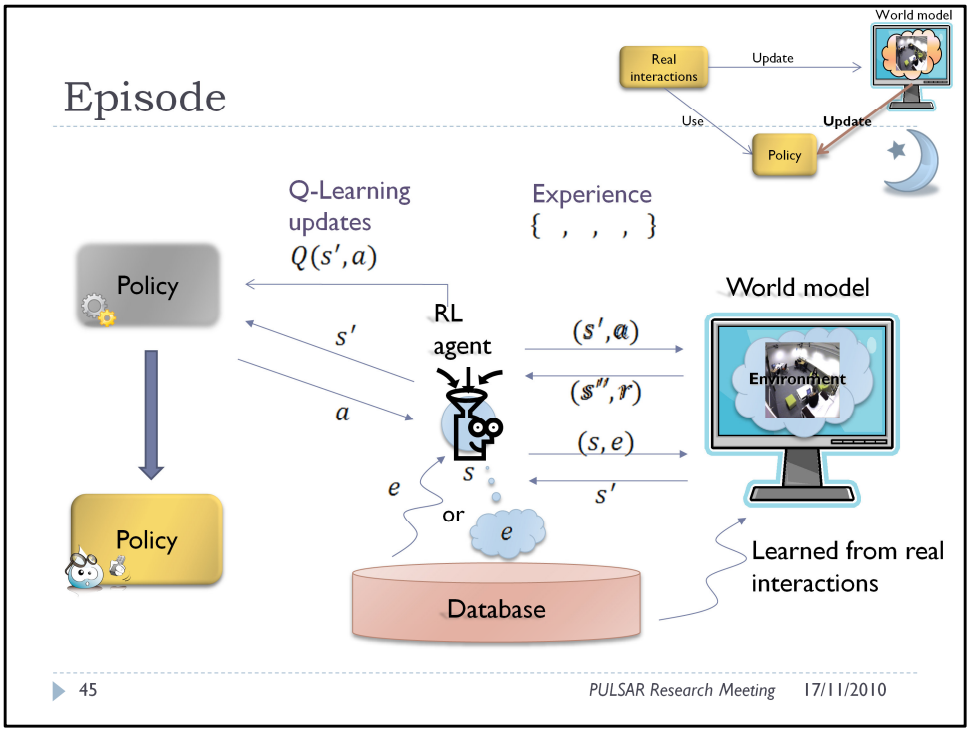
Exemples serviront à construire modèle du monde.

Partie interactive, sans apprentissage

Episode

- Episode steps have 2 stages:
  - Select an event that modifies the state
  - Select an action to react to that event





Partie non interactive, apprentissage

1 cycle = 1 itération

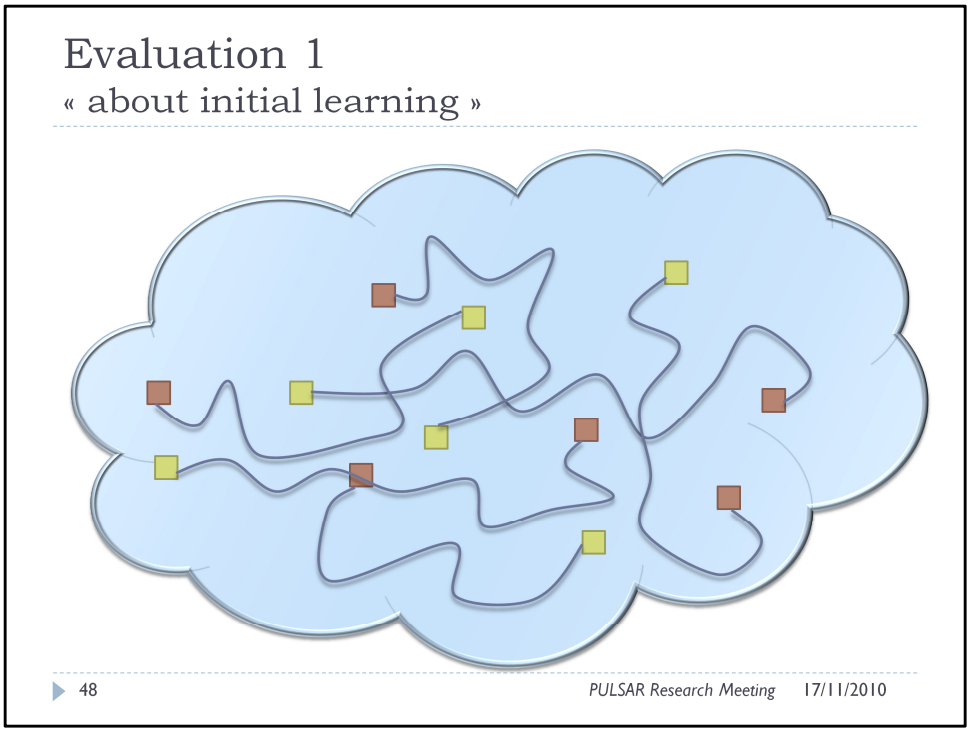
k itérations = 1 épisode

Outline

- Problem statement
- Learning in ubiquitous systems
- User Study
- Ubiquitous system
- Reinforcement learning of a context model
- **Experimentations and results**
- Conclusion

Experimentations

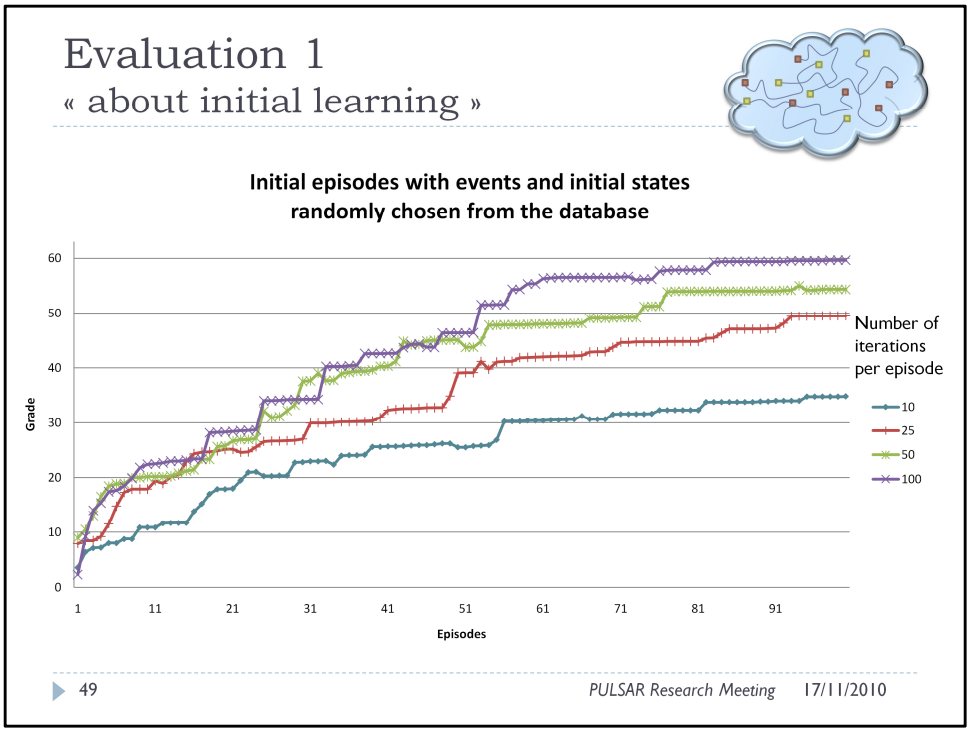
- General public survey → qualitative evaluation
- Quantitative evaluations in 2 steps:
  - Evaluation of the initial phase
  - Evaluation of the system during normal functioning



Supervisé : généralisation pas mémorisation d'exemples → permet exploration

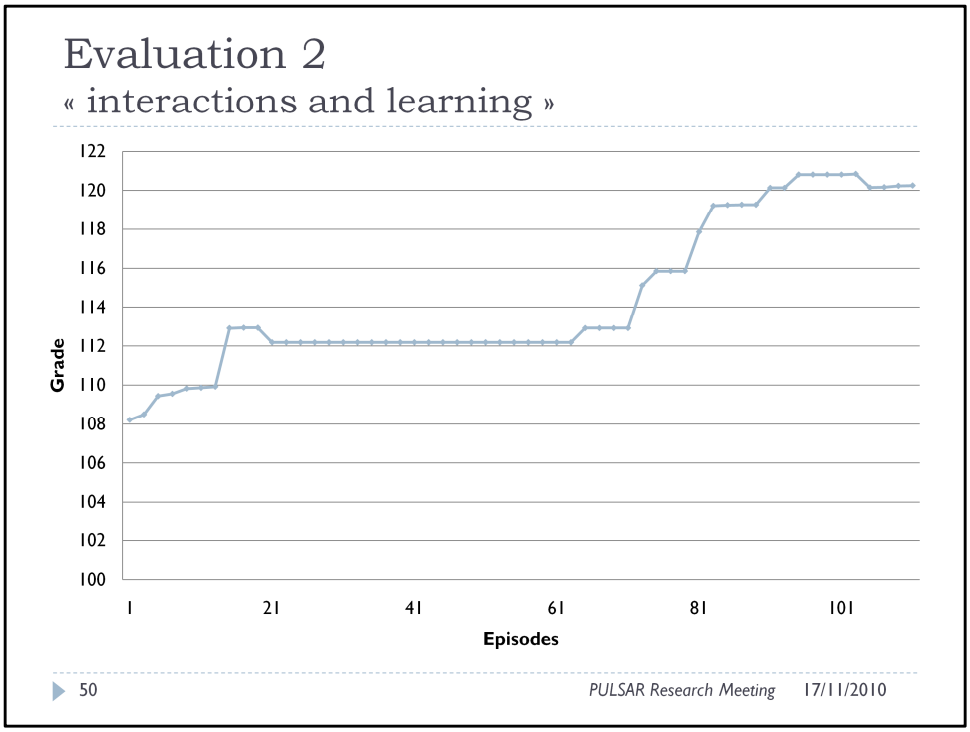
On a le modèle du monde initial, on veut l'explorer pour le traduire en un comportement initial. Il y a 4 paramètres à fixer : comment choisir les états de départ (carrés verts), comment choisir l'événement à chaque pas du chemin, combien de chemins faire, quelle longueur de chemins choisir ?
1 chemin = 1 épisode de k itérations.
Les 2 premiers ont été fixés dans le manuscrit (états de départ des chemins, et événements à chaque pas du chemin choisis aléatoirement parmi ceux déjà observés → permet d'obtenir un résultat optimal).
Pour la longueur et le nombre des chemins, voir le graphique suivant.

Exemple pour état de départ : l'état où Karl entre dans le bureau.



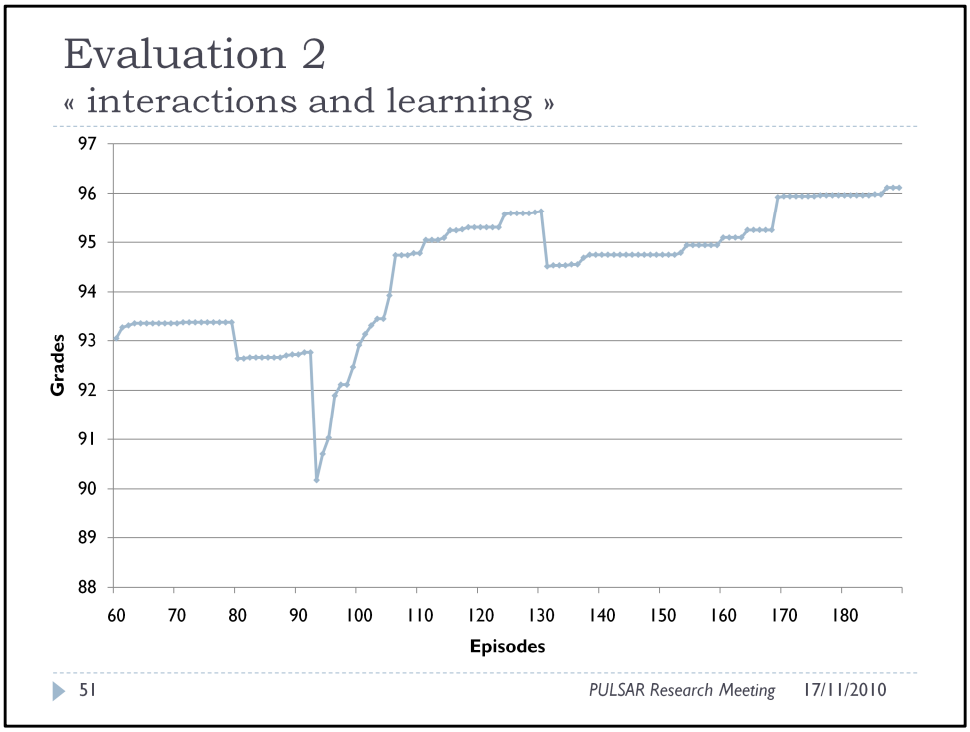
En abscisse : le nombre de chemins, en ordonnée : une note exprimant la ressemblance du comportement appris au comportement désiré (on a cette note car c'est une expérience et l'expérimentateur a donné des notes, dans la « vie réelle », la note n'est pas disponible). Les différentes courbes correspondent à différentes longueurs de chemins.

Dans l'absolu, on devrait choisir le dernier point de la courbe violette (100 itérations/épisode), mais dans la pratique on devrait choisir un compromis entre temps d'exécution et qualité du résultat.
Entre la courbe verte et la violette, on double le temps de calcul, et ce doublement n'est pas justifié par la si faible augmentation en termes de qualité de comportement. De même, à partir de l'épisode 50, on ne gagne pas grand-chose au niveau de la note, alors que le temps de calcul augmente toujours autant. Donc on pourra choisir, pour cette phase initiale, d'exécuter 50 épisodes de 50 itérations chacun.



Point de départ = comportement initial.
Puis l'expérimentateur interagit avec l'environnement. Ceci permet à l'assistant de voir des parties du monde non incluses dans le modèle initial, et donc l'apprendre. Là où la courbe est plate, l'assistant a appris tout ce qu'il pouvait, il n'observe rien de nouveau. Puis, dès qu'il observe une nouveauté, il continue à apprendre.

L'expérimentateur utilise le tableau de bord (cf backup slides) pour interagir avec l'environnement, simplement pour lui faciliter l'expérience, mais on a toujours le facteur humain dans l'expérience.



Là, l'expérimentateur a changé d'avis (vers x=100) → la note a baissé avant de remonter (exemple d'évolution de l'environnement).

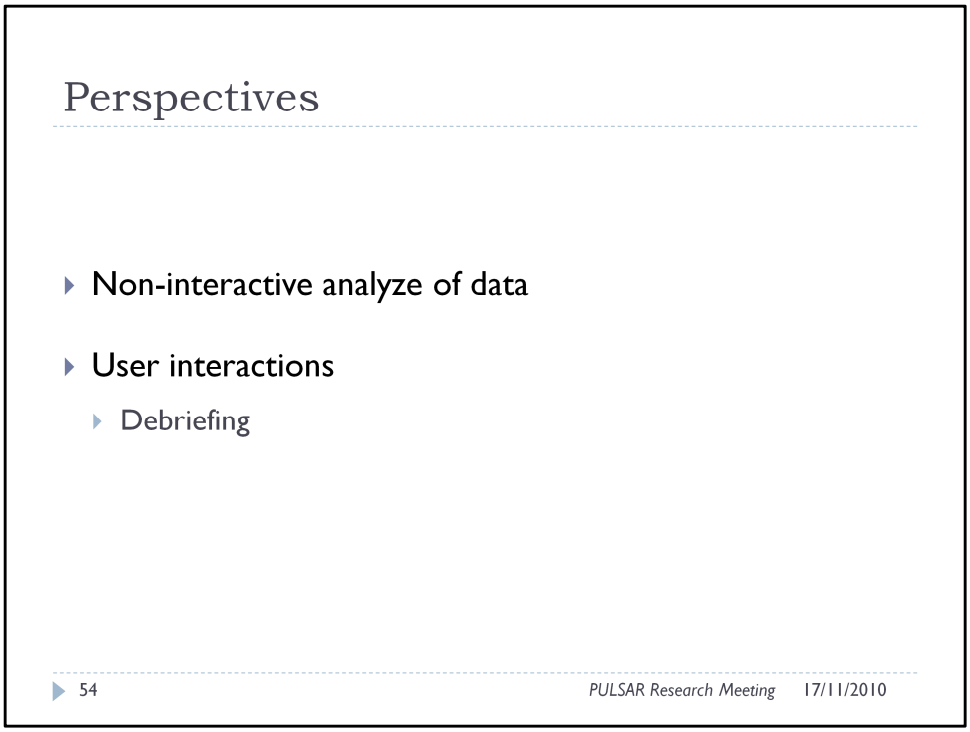
Par exemple, l'utilisateur ne veut plus lire ses mails le matin, mais l'après-midi.

Outline

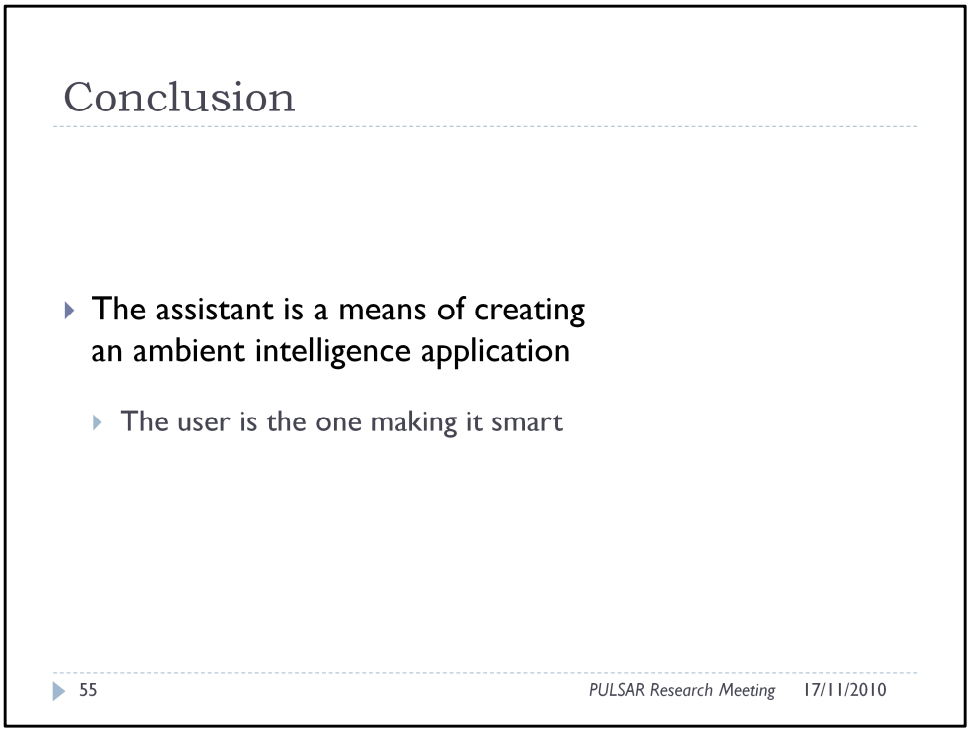
- Problem statement
- Learning in ubiquitous systems
- User Study
- Ubiquitous system
- Reinforcement learning of a context model
- Experimentations and results
- **Conclusion**

Contributions

- Personalization of a ubiquitous system
  - Without explicit specification
  - Easy to evolve
- Adaptation of indirect reinforcement learning to a real-world problem
  - Construction of a world model
  - Injection of initial knowledge
- Deployment of a prototype



Interactions : on n'est peut-être pas obligés de toujours tout découvrir par exploration. C'est plus efficace pour nous de pouvoir parfois demander à l'utilisateur, et lui, il préfère aussi parfois pouvoir donner son avis (cf enquête).



L'assistant est une boite blanche, il n'est pas intelligent par sa conception, ce sont ses interactions avec l'utilisateur, les retours qu'il donne, qui rendent l'assistant intelligent. C'est l'engagement de l'utilisateur dans le système qui lui donne de la valeur ajoutée.

Thanks for your attention

Questions?

Bibliography

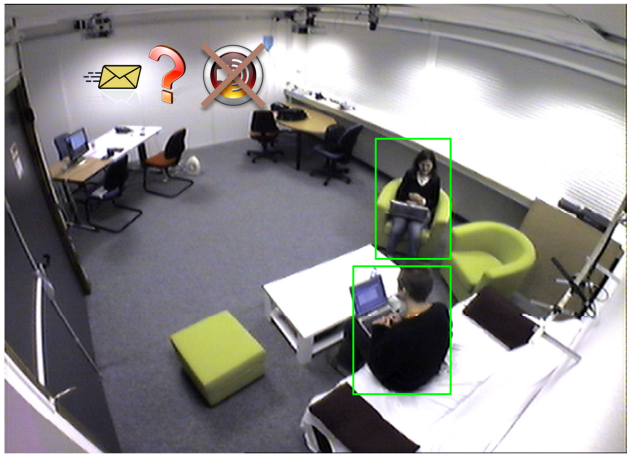
[Bellotti and Edwards, 2001]	Victoria BELLOTTI and Keith EDWARDS. « Intelligibility and accountability: human considerations in context-aware systems ». *In Human-Computer Interaction*, 2001.
[Brdiczka *et al.*, 2007]	Oliver BRDICZKA, James L. CROWLEY and Patrick REIGNIER. « Learning Situation Models for Providing Context-Aware Services ». *In Proceedings of HCI International*, 2007.
[Buffet, 2003]	Olivier Buffet. « Une double approche modulaire de l'apprentissage par renforcement pour des agents intelligents adaptatifs ». Thèse de doctorat, Université Henri Poincaré, 2003.
[Emonet *et al.*, 2006]	Rémi Emonet, Dominique Vaufreydaz, Patrick Reignier and Julien Letessier. « O3MiSCID: an Object Oriented Opensource Middleware for Service Connection, Introspection and Discovery ». *In IEEE International Workshop on Services Integration in Pervasive Environments*, 2006.
[Kaelbling, 2004]	Leslie Pack Kaelbling. « Life-Sized Learning ». Lecture at CSE Colloquia, 2004.
[Maes, 1994]	Pattie MAES. « Agents that reduce work and information overload ». *In Commun. ACM*, 1994.
[Maisonnasse 2007]	Jerome MAISONNASSE, Nicolas GOURIER, Patrick REIGNIER and James L. CROWLEY. « Machine awareness of attention for non-disruptive services ». *In HCI International*, 2007.
[Moore, 1975]	Gordon E. MOORE. « Progress in digital integrated electronics ». *In Proc. IEEE International Electron Devices Meeting*, 1975.

Bibliography

[Nonogaki and Ueda, 1991]	Hajime Nonogaki and Hirotada Ueda. « FRIEND21 project: a construction of 21st century human interface ». *In CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, 1991.
[Roman *et al.*, 2002]	Manuel ROMAN, Christopher K. HESS, Renato CERQUEIRA, Anand RANGANATHAN, Roy H. CAMPBELL and Klara NAHRSTEDT. « Gaia: A Middleware Infrastructure to Enable Active Spaces ». *In IEEE Pervasive Computing*, 2002.
[Sutton, 1991]	Richard S. Sutton. « Dyna, an integrated architecture for learning, planning, and reacting ». *In SIGART Bull,* 1991.
[Weiser, 1991]	Mark WEISER. « The computer for the 21st century ». *In Scientic American*, 1991.
[Weiser, 1994]	Mark WEISER. « Some computer science issues in ubiquitous computing ». *In Commun. ACM*, 1993.
[Weiser et Brown, 1996]	Mark WEISER and John Seely BROWN. « The coming age of calm technology ». http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm, 1996.
[Watkins, 1989]	CJCH Watkins. « Learning from Delayed Rewards ». Thèse de doctorat, University of Cambridge, 1989.

Intelligence ambiante

- **Context-aware computing**



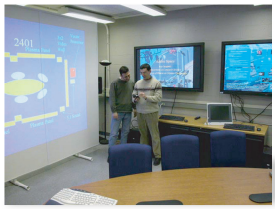
1. Perception
2. Décision

Intelligence = Percevoir l'environnement et la situation
Sélectionner et exécuter une action adéquate qui
Modifie l'environnement
ou bien
Modifie le système lui-même
Afin de se trouver dans un état désirable (par rapport à un certain but ou critère).

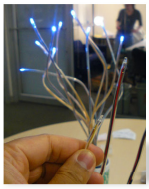
Intelligence dans un système ubiquitaire :
Percevoir la situation de l'utilisateur (le **contexte**)
Lui rendre un service adéquat

État de l'art

[Nonogaki et Ueda, 1991]
FRIEND21



[Roman *et al.*, 2002]
Gaia



Blossom

Sajid SADI et Pattie MAES

http://consciousanima.net/projects/blossom/

60 *PULSAR Research Meeting* 17/11/2010

FRIEND21 : recherche sur les principes à respecter pour réaliser les interfaces du 21ème siècle. Communication et accès à l'information facilités et prise en compte du contexte.

Gaia : un système d'exploitation gérant un « espace actif », en prenant en compte le contexte. Gère les incertitudes de la perception par apprentissage mais pas d'apprentissage des préférences. Interaction transparente étendue sur tous les dispositifs de l'environnement.

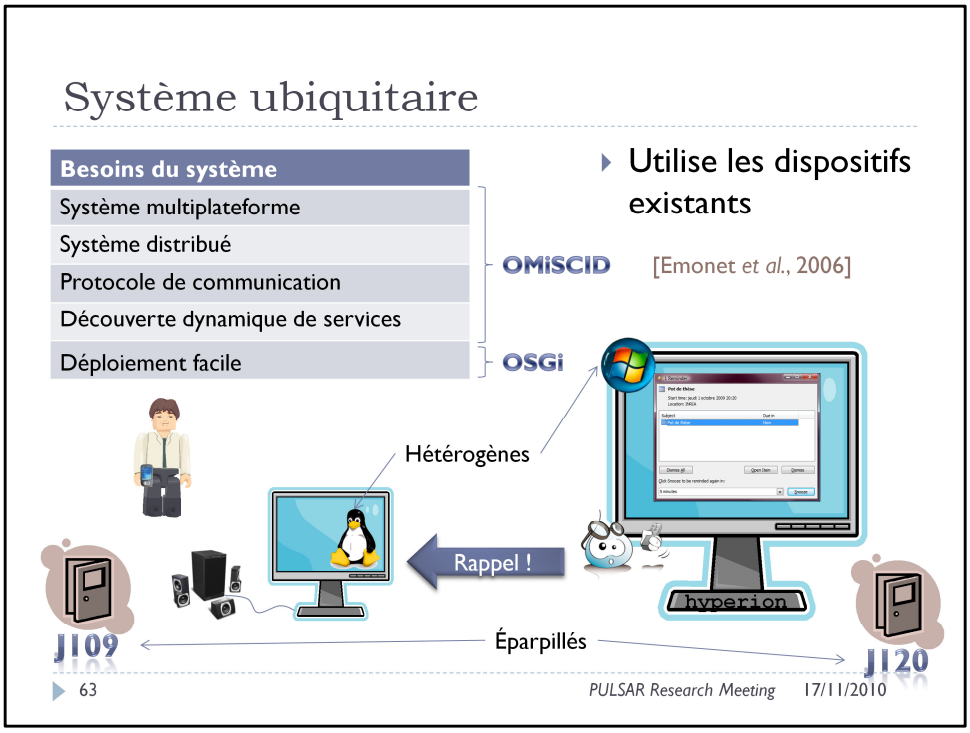
Blossom (MediaLab du MIT) : permet de garder le contact avec ses proches de manière non-intrusive, en se focalisant sur la présence en périphérie de l'attention plutôt que sur une communication directe.

Personnalisation

- Personnalisation d'un agent informatique complexe qui assiste l'utilisateur.
- Deux solutions [Maes, 1994]
  - L'utilisateur spécifie lui-même le comportement
    - Système trop complexe ⇨ Tâche laborieuse
    - Peu-évolutif
  - Choix prédéfini par un expert
    - Non-personnalisé
    - Non-évolutif
    - Utilisateur ne maîtrise pas tout le système

Plan

- Présentation du problème
- Apprentissage dans les systèmes ubiquitaires
- **Enquête grand public**
- Système ubiquitaire
- Apprentissage par renforcement du modèle de contexte
- Expérimentations et résultats
- Conclusion



Besoin de déploiement facile car on veut un système qui marche ne permanence (système vivant en permanence), on veut jamais l'arrêter pour maintenance.
→ Déploiement à chaud
→ Administration à distance
→ Gestion des modules à partir d'un dépôt central → pratique pour mettre à jour et pour ajouter dynamiquement de nouvelles fonctionnalités sans intervention manuelle.

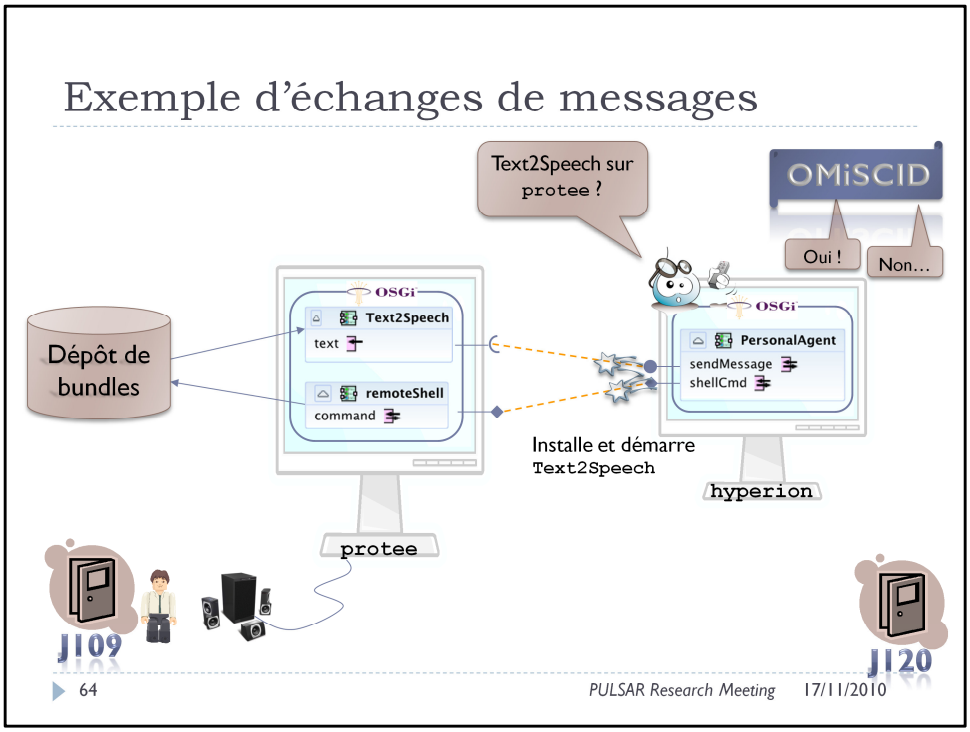
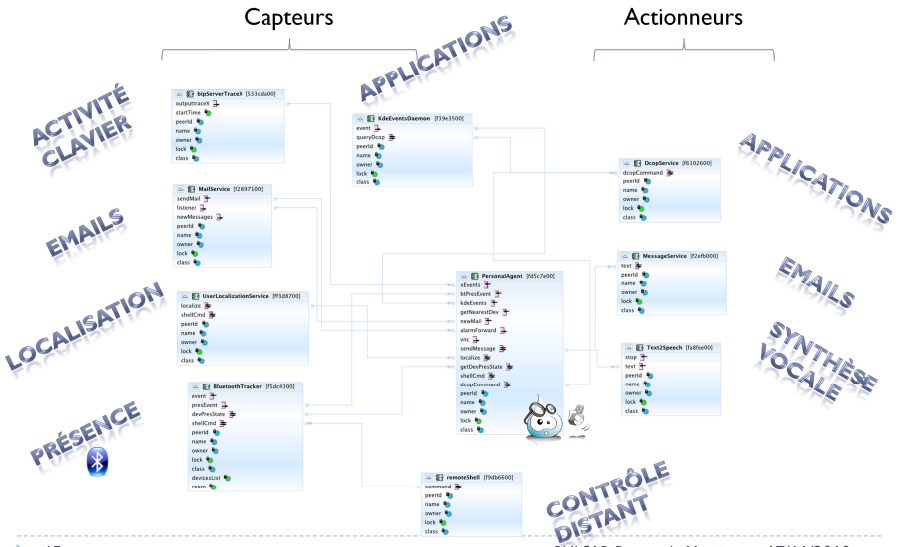
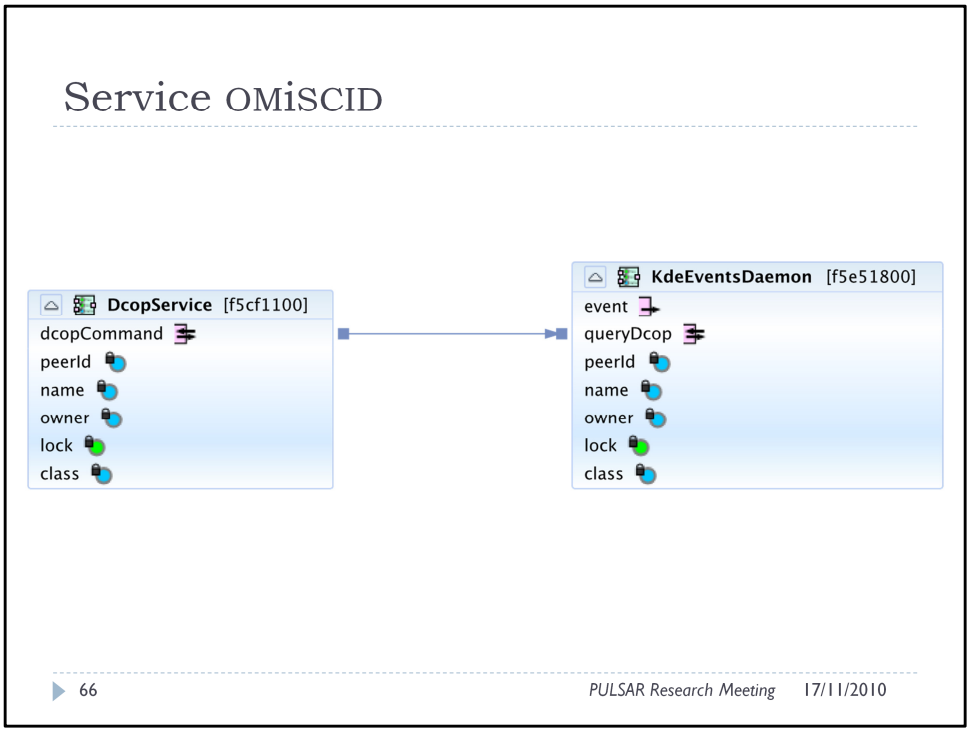


Illustration du déploiement à chaud (stratégie opportuniste) :
Exemple de transmission du rappel vu à un niveau plus bas (implémentation).

Le module AssistantPersonnel (« PersonalAgent ») s'exécute sur la machine de bureau de l'utilisateur. Il veut contacter le module « Text2Speech » (module de synthèse vocale) sur la machine « protee » (celle connectée aux haut-parleurs du bureau dans lequel se trouve l'utilisateur). C'est OMiSCID qui permet de brancher les modules les uns aux autres. Dans un 1er temps, OMiSCID répond à l'assistant que Text2Speech n'est pas en exécution sur protee. Alors l'assistant demande à OMiSCID d'obtenir une référence vers remoteShell (administration à distance) sur protee. On suppose que remoteShell doit toujours être disponible. L'assistant se branche sur remoteShell et lui envoie une commande pour installer et démarrer Text2Speech. remoteShell le fait à partir du dépôt central. Maintenant l'assistant redemande à OMiSCID une référence vers Text2Speech sur protee et l'obtient. Il s'y branche et lui envoie le texte à prononcer.

Interconnexion des modules

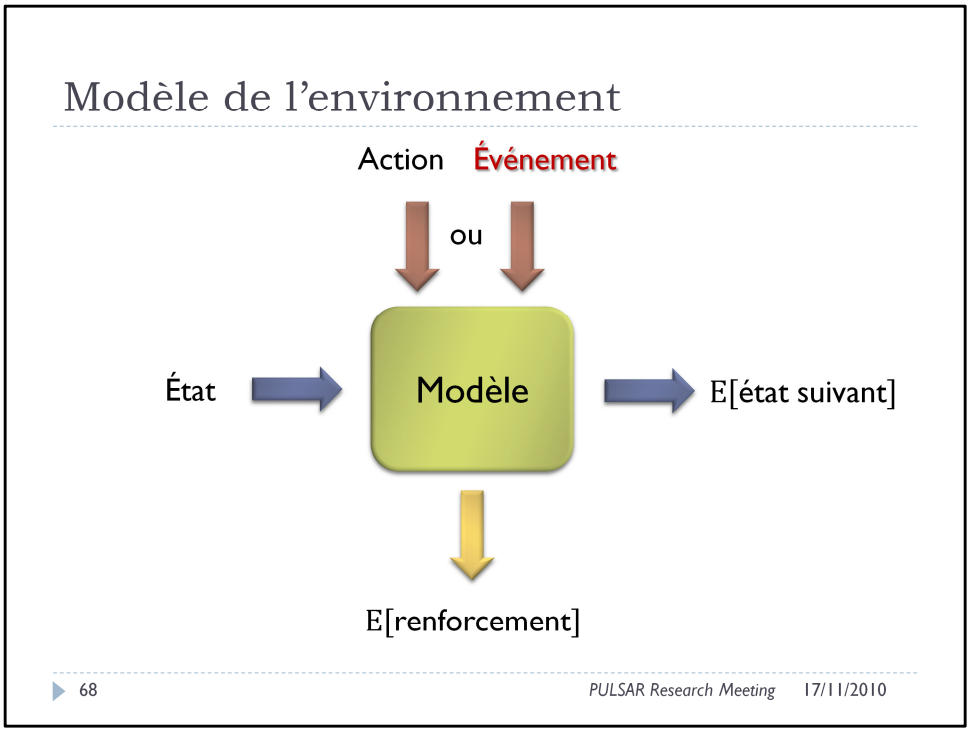




Les modules du système sont implémentés en tant que « services OMiSCID »
Ils exposent des connecteurs et des variables
    Entrée / sortie / entrée-sortie pour les connecteurs
    Lecture / lecture-écriture pour les variables
Deux services communiquent en « branchant » deux de leurs connecteurs

Définition d'un état

Prédicat	Arguments
alarm	title, hour, minute
xActivity	machine, isActive
inOffice	user, office
absent	user
hasUnreadMail	from, to, subject, body
entrance	isAlone, friendlyName, btAddress
exit	isAlone, friendlyName, btAddress
task	taskName
user	login
userOffice	office, login
userMachine	machine, login
computerState	machine, isScreenLocked, isMusicPaused



Normalement il n'y a que les actions qui modifient l'état. L'utilisateur ne fait pas partie du monde, donc n'est pas modélisé dans le modèle du monde, donc les événements qui sont provoqués par ses actions sont des entrées du modèle.

Réduction de l'espace d'états

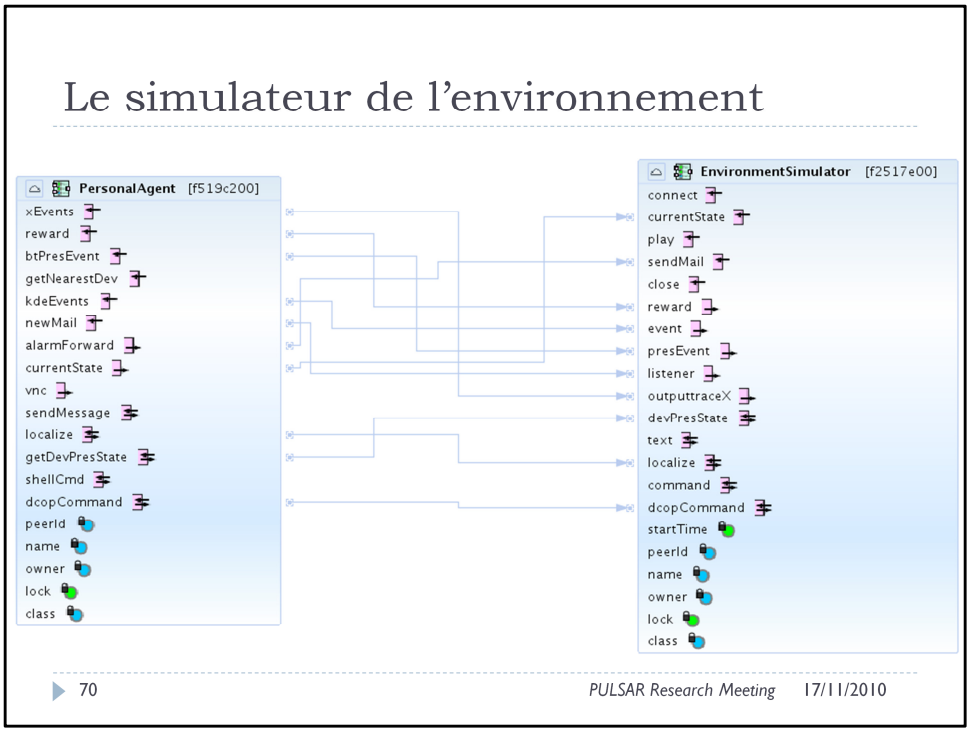
- Accélération de l'apprentissage
  - Factorisation d'états

Jokers
<*> et <+>

État	Action	Q-valeur
…entrance(isAlone=true, friendlyName=<+>, btAddress=<+>)…	pauseMusic	125.3

  - Division d'états

État	Action	Q-valeur
…hasUnreadMail(from=**boss**, to=<+>, subject=<+>, body=<+>)…	inform	144.02
…hasUnreadMail(from=**newsletter**, to=<+>, subject=<+>, body=<+>)…	notInform	105



Le simulateur contient une simulation de tous les capteurs et effecteurs en définissant les mêmes connecteurs qu'eux tous.
Les messages échangés sont de même format que sur les vrais connecteurs. Les échanges de messages suivent le même format (ex: requête – réponse ou juste requête etc.)

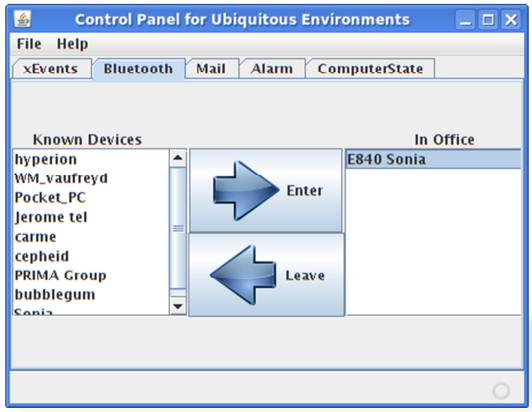
Critère d'évaluation : la note

- Résultat de l'AR : une Q-table
- Comment savoir si elle est « bonne » ?
- Apprentissage réussi si
  - Comportement correspond aux souhaits de l'utilisateur
  - Et c'est mieux si on a beaucoup exploré et si on a une estimation du comportement dans beaucoup d'états

$$note = \frac{1}{13}(10 \times n_{correct} + 2 \times p_{nonNul} + n_{total})$$

« Le tableau de bord »

- Permet d'envoyer par un clic les mêmes événements que les capteurs



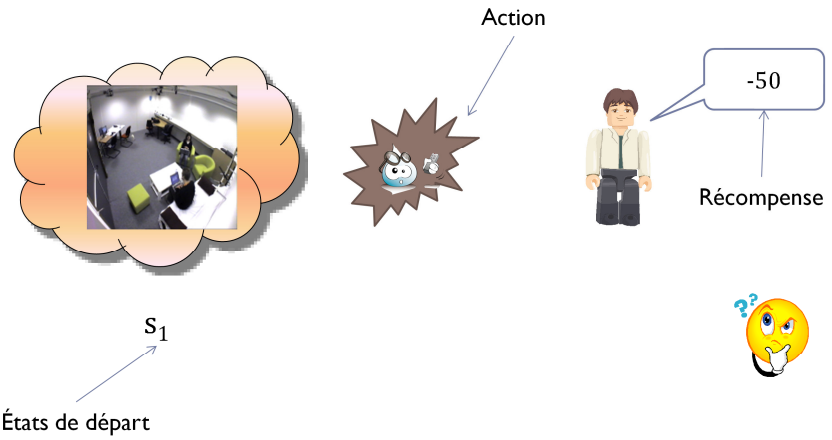
Modèle de récompense

- Ensemble d'entrées spécifiant
  - Des contraintes sur certains arguments de l'état
  - Une action
  - La récompense

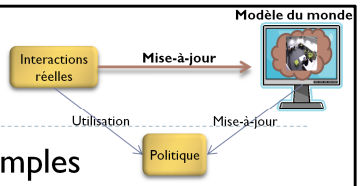
Modèle de récompense

Modèle de transition

Modèle de récompense



Apprentissage supervisé du modèle de récompense



- La base de données contient des exemples *{état précédent, action, récompense}*