CS-639 — Interaction Design Studio

AI as Design Material — Proactivity & Timing

Professor Bilge Mutlu

Today

— **W08 recap:** You learned to choose **how much** a system should do (Parasuraman)

— **This week:** We ask **when and why** it should act

— The right action at the wrong time is the wrong action

Agency tells you the level. Timing tells you the moment.

W08 → W09: From How Much to When

	W08: Agency	W09: Proactivity
Core question	How much should the system do?	When and why should it act?
Framework	Parasuraman et al. (2000) — 10 levels, 4 stages	Horvitz (1999) — expected utility, 12 principles
Design tool	Four decision factors + HAX	Four designer questions + Act/Ask/Wait
Key tension	Control vs. convenience	Helpfulness vs. interruption
Material properties	Adaptivity, Delegation	Initiative, Inference

Part 1: The Timing Question

When should the system act — and what happens if it gets the timing wrong?

Horvitz (1999)

The foundational framework for proactive intelligent systems.[1]

"Ideal action under uncertainty hinges on a consideration of expected utility — balancing the expected costs and benefits of taking actions."

— **12 principles** for mixed-initiative interaction

— Tested in **LookOut** — an email assistant that decided when to help schedule meetings

[1] Horvitz (1999), "Principles of Mixed-Initiative User Interfaces," CHI '99

Principles of Mixed-Initiative User Interfaces

Eric Horvitz
Microsoft Research
Redmond, WA 98025 USA
+1 425 936 2127
horvitz@microsoft.com

ABSTRACT
Recent debate has centered on the relative promise of focusing user-interface research on developing new metaphors and tools that enhance users' abilities to directly manipulate objects *versus* directing effort toward developing interface agents that provide automation. In this paper, we review principles that show promise for allowing engineers to enhance human–computer interaction through an elegant coupling of automated services with direct manipulation. Key ideas will be highlighted in terms of the LookOut system for scheduling and meeting management.

Keywords
Intelligent agents, direct manipulation, user modeling, probability, decision theory, UI design

INTRODUCTION
There has been debate among researchers about where great opportunities lay for innovating in the realm of human—computer interaction [10]. One group of researchers has expressed enthusiasm for the development and application of new kinds of automated services, often referred to as interface "agents." The efforts of this group center on building machinery for sensing a user's activity and taking automated actions [4,5,6,8,9]. Other researchers have suggested that effort focused on automation might be better expended on exploring new kinds of metaphors and conventions that enhance a user's ability to *directly manipulate* interfaces to access information and invoke services [1,13]. Innovations on both fronts have been fast paced. However, there has been a tendency for a divergence of interests and methodologies versus focused attempts to leverage innovations in both arenas.

We have pursued principles that provide a foundation for integrating research in direct manipulation with work on interface agents. Our goal is to avoid focusing solely on one tack or the other, but to seek valuable synergies between the two areas of investigation. Surely, we should avoid building complex reasoning machinery to patch fundamentally poor designs and metaphors. Likewise, we wish to avoid limiting designs for human–computer interaction to direct manipulation when significant power and efficiencies can be gained with automated reasoning. There is great opportunity for designing innovative user interfaces, and new human–computer interaction modalities by considering, from the ground up, designs that take advantage of the power of direct manipulation and potentially valuable automated reasoning [2].

PRINCIPLES FOR MIXED-INITIATIVE UI
Key problems with the use of agents in interfaces include poor guessing about the goals and needs of users, inadequate consideration of the costs and benefits of automated action, poor timing of action, and inadequate attention to opportunities that allow a user to guide the invocation of automated services and to refine potentially suboptimal results of automated analyses. In particular, little effort has been expended on designing for a *mixed-initiative* approach to solving a user's problems—where we assume that intelligent services and users may often collaborate efficiently to achieve the user's goals.

Critical factors for the effective integration of automated services with direct manipulation interfaces include:

(1) **Developing significant value-added automation.** It is important to provide automated services that provide *genuine value* over solutions attainable with direct manipulation.

(2) **Considering uncertainty about a user's goals.** Computers are often uncertain about the goals and current the focus of attention of a user. In many cases, systems can benefit by employing machinery for inferring and exploiting the uncertainty about a user's intentions and focus.

(3) **Considering the status of a user's attention in the timing of services.** The nature and timing of automated services and alerts can be a critical factor in the costs and benefits of actions. Agents should employ models of the attention of users and consider the *costs and benefits of deferring action* to a time when action will be less distracting.

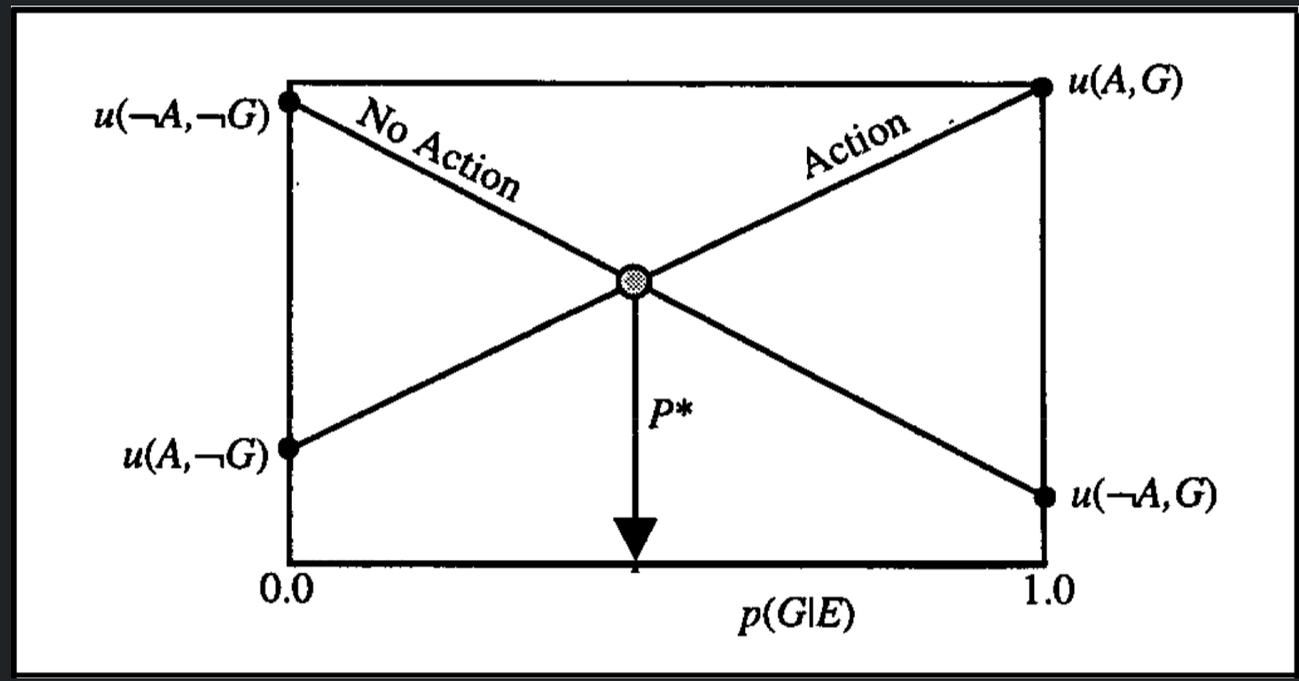
(4) **Inferring ideal action in light of costs, benefits, and uncertainties.** Automated actions taken under uncertainty in a user's goals and attention are associated with context-dependent costs and benefits.

Expected Utility: The Core Insight

The system should act when:

$$Benefit \times P(correct) > Cost \times P(incorrect)$$

Every proactive system implicitly makes this calculation — from autocorrect to fall detection.



Part 2: Horvitz's 12 Principles as Four Designer Questions

A practical framework for deciding when to act

Horvitz's 12 Principles

Principle	Description
P01 — Developing significant value-added automation	Act only when automation adds clear value
P02 — Considering uncertainty about a user's goals	Maintain representations of uncertainty
P03 — Considering the status of a user's attention	Infer attentional focus; time actions accordingly
P04 — Inferring ideal action in light of costs/benefits	Weigh expected costs and benefits of acting
P05 — Employing dialog to resolve key uncertainties	Use efficient dialog when uncertain
P06 — Allowing efficient direct invocation/termination	Users can invoke, pause, resume, cancel
P07 — Minimizing cost of poor guesses	Design to minimize damage of errors
P08 — Scoping precision of service to match uncertainty	Narrow scope when less certain
P09 — Providing mechanisms for efficient collaboration	Support correction and refinement
P10 — Employing social conventions for interaction	Respect social norms and timing
P11 — Maintaining memory of interaction	Remember past interactions
P12 — Learning from user behavior	Improve from user feedback over time

12 Principles → Four Designer Questions

1. Is it worth acting?

— P1: Value-added automation
— P4: Costs and benefits

2. Do I know enough?

— P2: Uncertainty about goals
— P5: Dialog to resolve uncertainty
— P8: Scope precision

3. Is the timing right?

— P3: Status of attention
— P7: Cost of poor guesses
— P10: Social conventions

4. Can the user stay in control?

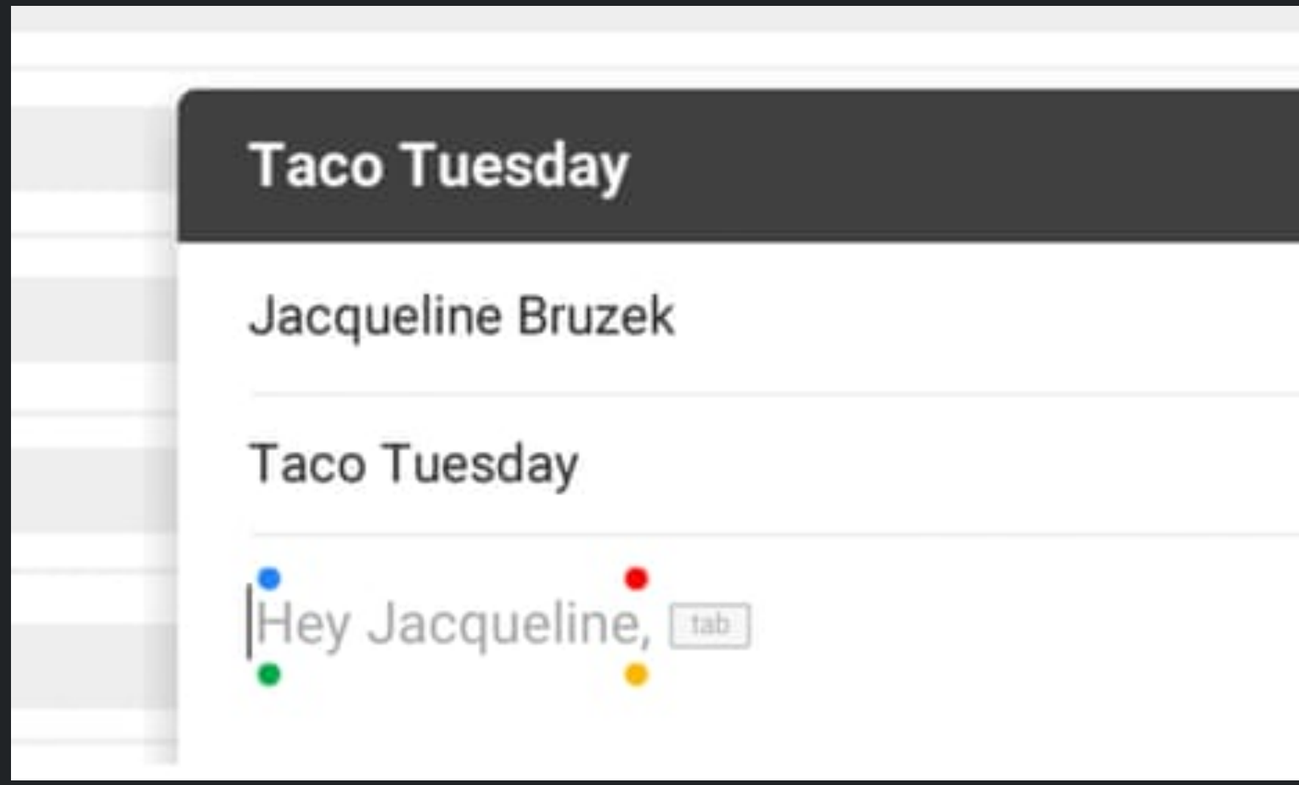
— P6: Direct invocation/termination
— P9: Efficient collaboration
— P11: Memory of interaction
— P12: Learning from behavior

Q1: Is It Worth Acting? — P1, P4

Act only when expected value exceeds expected cost of interruption.

Gmail Smart Compose: Gray ghost text as you type. Benefit: saves keystrokes. Cost if wrong: near zero — just keep typing. Threshold: very low.

Apple Watch Fall Detection: Calls 911 after 60s. Benefit: life-saving. Cost if wrong: embarrassing. Threshold: high — but stakes justify it.



Q2: Do I Know Enough? — P2, P5, P8

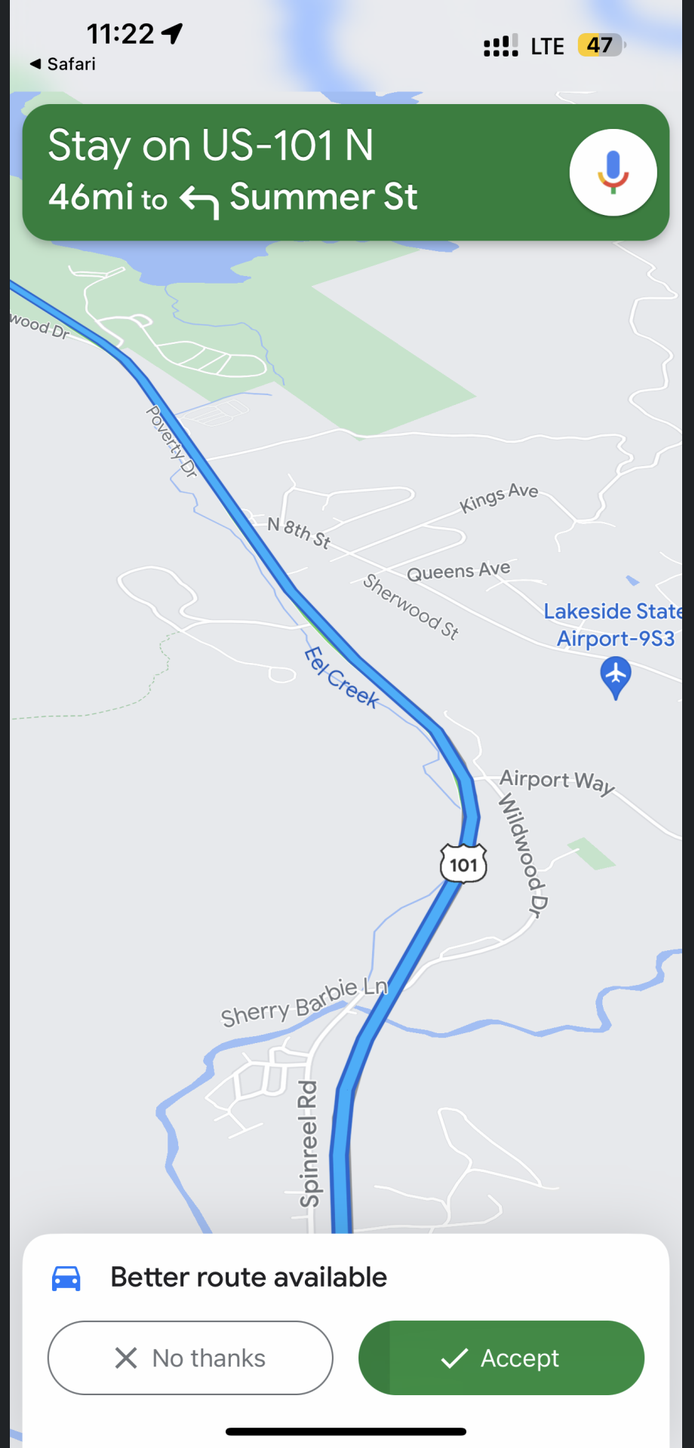
When uncertain, narrow scope, ask, or wait for more signal.

Google Maps — graduated confidence:

— **High** → Auto-reroutes around a road closure (acts)

— **Medium** → "Faster route — save 5 min?" (asks)

— **Low** → Shows traffic overlay, no suggestion (waits)

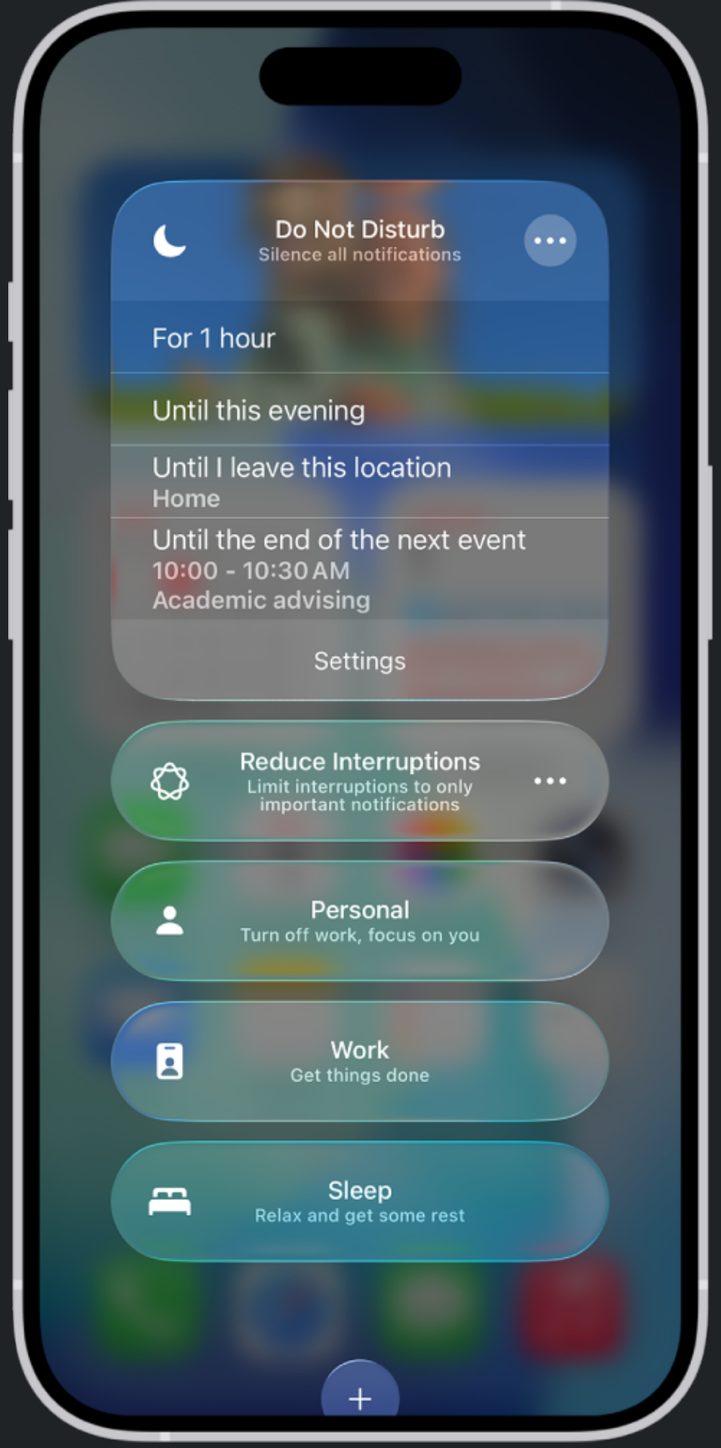


Q3: Is the Timing Right? — P3, P7, P10

The right action becomes wrong if the timing disrupts focus or social context.

iOS Focus Modes — models user attention: silences social apps at work, allows only emergency contacts during sleep.

Slack — models social context: "It's 2:30 AM for this person. Schedule for their morning?"



Q4: Can the User Stay in Control? — P6, P9, P11, P12

Horvitz's control principles (P6, P9, P11, P12) in action form a cycle: invoke → dismiss → correct → teach.

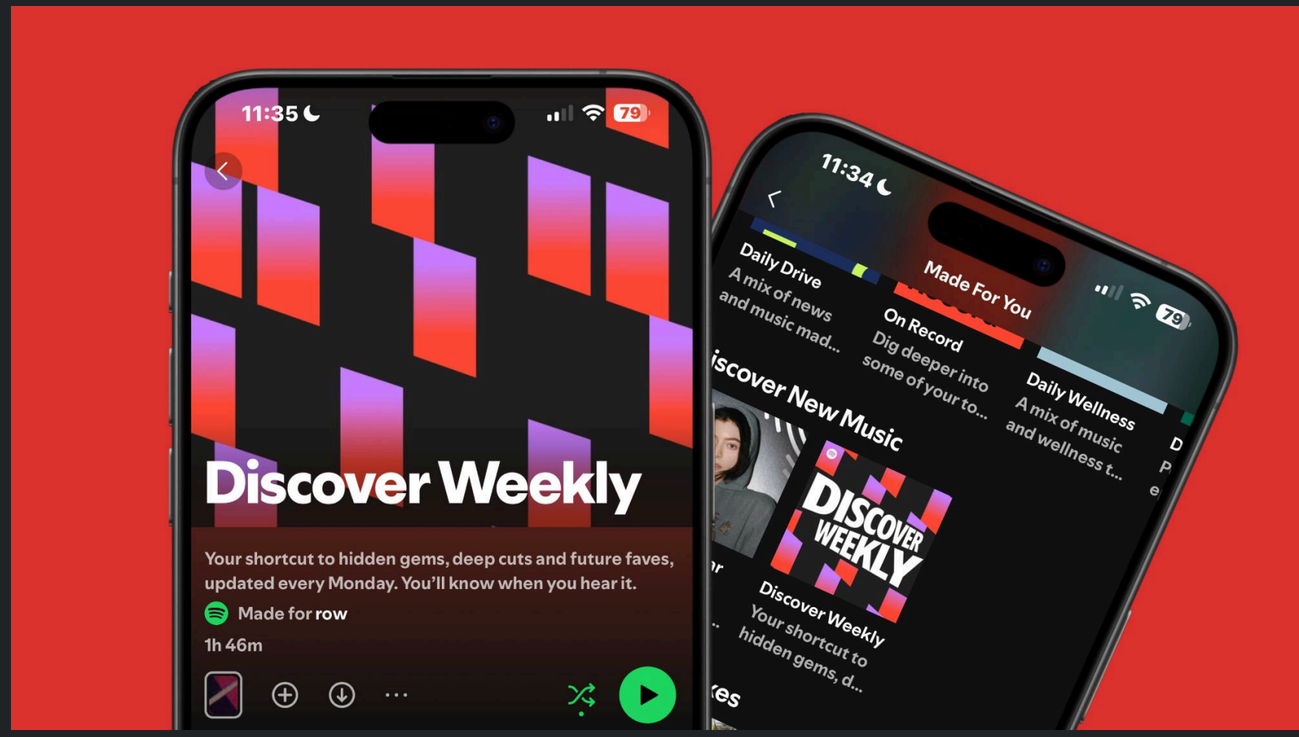
Spotify Discover Weekly:

— **Invoke** (P6) — playlist appears every Monday

— **Dismiss** (P6) — skip a song or the whole playlist

— **Correct** (P9) — like/dislike adjusts recommendations

— **Teach** (P11, P12) — listening history refines the model



The Four Questions — Summary Checklist

Before your proactive system acts, ask:

Question	If "no" →	Horvitz Principles
1. Is it worth acting?	Don't act — the cost exceeds the benefit	P1, P4
2. Do I know enough?	Ask, narrow scope, or wait for more signal	P2, P5, P8
3. Is the timing right?	Wait — surface the action later	P3, P7, P10
4. Can the user stay in control?	Add invoke/dismiss/correct/ teach mechanisms	P6, P9, P11, P12

If the answer to all four is "yes," act. If any answer is "no," ask or wait.

Part 3: Act, Ask, or Wait

Three options for every proactive moment — and what determines the right one

Three Options

Horvitz's expected utility analysis divides the decision space into three regions — autonomous action, dialog, or inaction — based on two threshold probabilities. We'll call these **Act, Ask, and Wait:**

Option	Horvitz's term	What the system does	When to use
ACT	Autonomous action	Executes without asking	High confidence, low cost, high frequency
ASK	Dialog	Presents suggestion; user decides	Medium confidence, moderate cost
WAIT	Inaction	Gathers information silently	Low confidence, high cost, or bad timing

The choice isn't permanent — the same system should shift between Act, Ask, and Wait depending on context.

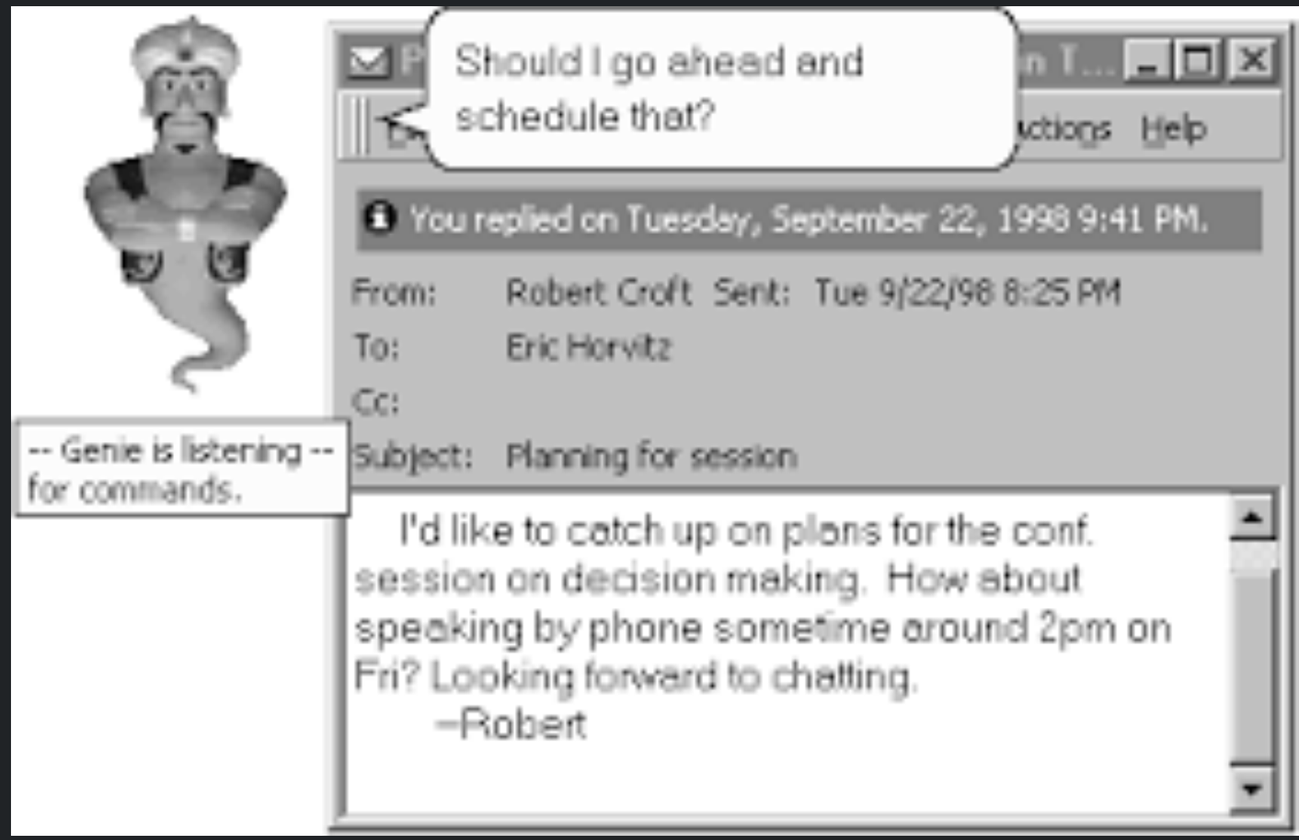
LookOut: Horvitz's Case Study

A research prototype Outlook plugin that watched for meeting requests and decided how to respond:

— **High confidence** → open calendar with pre-filled event (ACT)

— **Medium confidence** → "Schedule this?" (ASK)

— **Low confidence** → do nothing (WAIT)




Microsoft

LookOut's Legacy

LookOut never shipped — it was a Microsoft Research prototype. But the pattern it established shows up everywhere today:

— Gmail's event detection from email
— Apple's Siri Suggestions
— Google Calendar's auto-add from email

The prototype didn't survive; **the design pattern did.**

Act, Ask, or Wait — Modern Examples

Product	ACT	ASK	WAIT
Gmail	Spam filter moves messages (99.9% confidence)	Smart Reply suggests three responses	Smart Compose waits until you start typing
Google Maps	Auto-reroutes around closure	"Faster route — save 5 min?"	Shows traffic overlay, no suggestion
Apple Watch	Fall detection calls 911 after 60s	"It looks like you fell. Are you OK?"	Monitors movement patterns silently
Autocorrect	Fixes "teh" → "the" instantly	Shows suggestion bubble for unusual words	Learns your vocabulary over time

Notice: these aren't four different systems — they're four systems that shift between modes based on confidence and context.

What Shifts the Threshold?

The boundary between Act, Ask, and Wait shifts based on five factors — drawn from Horvitz (1999) and Parasuraman et al. (2000):

Factor	Shifts toward ACT	Shifts toward WAIT	Source
User attention	Idle or low-focus	Deep focus	Horvitz P3
Error cost	Cheap and reversible	Costly or permanent	Parasuraman
Time pressure	Urgent — delay has consequences	No rush	Horvitz P4, P7
User history	Strong user model	New user, no data	Horvitz P11, P12
Implementation constraints	Space for suggestions (e.g., screen real estate)	Limited display	—

The same feature might ACT for an expert user and ASK for a new one.

Designing the Ask: Levels of Initiative

When the system chooses to ASK, the ask itself has levels — recall Parasuraman et al.'s (2000) Levels 2–5 from W08:

— **Suggest** — "Faster route available" (Level 2: offer alternatives)

— **Recommend** — "Take I-90" pre-selected (Level 4: suggest one)

— **Prompt** — "Add to calendar?" (Level 5: execute if approved)

— **Negotiate** — "How about Tuesday?" (user counter-proposes) — Allen et al. (1999)

Part 4: Material Properties

Connecting proactivity to the intelligence design framework

Initiative: How Does the System Take Action?

— **Passive** — responds only when asked (Siri waiting for "Hey Siri")

— **Reactive** — responds to triggers (spell check underlining)

— **Proactive** — anticipates needs (Maps suggesting departure)

— **Autonomous** — acts without involvement (Apple Watch fall detection)

Inference: How Does the System Interpret Context?

The system derives implicit needs from explicit actions (W07). More inference → less user effort, but more room for error:

Less inference ← → More inference

— Search returns exact keyword matches
— Google completes "best res" → "best restaurants near me open now"
— Spotify prepares a playlist before you ask

Higher inference enables higher initiative — but also creates more room for error. Horvitz's Q2 ("Do I know enough?") is really asking: is my inference reliable enough?

Connecting W08 + W09

Framework	W08: Agency	W09: Proactivity
Academic source	Parasuraman et al. (2000)	Horvitz (1999)
Core question	How much should the system do?	When should the system act?
Design tool	Four factors + HAX checklist	Four questions + Act/Ask/Wait
Material properties	Adaptivity, Delegation	Initiative, Inference
Together	Choose the right **level**	Choose the right **moment**

A well-designed intelligent system needs both: the right level of agency AND the right timing.

This Week

Reflection, studio challenge, and preparation for Wednesday

Reflection: Initiative Safari

Due before Wednesday | Graded: ✓ / ✓- / ✓+

Find **3 examples of proactive system behavior** in your daily life between now and Wednesday. For each one:

1. **What happened** — one sentence describing the proactive behavior
2. **Act, Ask, or Wait?** — classify the system's initiative level
3. **Four designer questions** — Is it worth acting? Do I know enough? Is the timing right? Can the user stay in control?
4. **One-sentence redesign** — suggest one change to improve the timing, initiative level, or control mechanism

Submit a **screenshot with annotations** and your answers for each example.

This primes Wednesday's studio — you'll design a proactive feature using the same framework you apply here.

Wednesday Studio: Designing Proactive Features

Design the **same proactive feature at three initiative levels** — Act, Ask, and Wait — then evaluate each with the four designer questions and design the threshold logic.

- **Pick a domain and a proactive feature** — your A1 app works well
- **Design at three levels:** Act (system executes), Ask (system suggests), Wait (system observes)
- **Evaluate with four designer questions:** Is it worth acting? Do I know enough? Is the timing right? Can the user stay in control?
- **Design the threshold:** What triggers transitions between Act, Ask, and Wait?

The test: if you can't specify what triggers a shift from Wait to Ask to Act, your threshold logic isn't specific enough.

Before Wednesday

— **Observe:** Notice proactive features in the apps you use — when do they act, ask, or wait?

— **Submit:** Initiative Safari reflection on Canvas (3 examples with screenshots)

— **Choose:** Your domain and proactive feature for Wednesday's studio

— **Think about:** What would change if your favorite proactive feature shifted one level?

Come ready to design the same feature at three initiative levels.

References

Core Framework:

— Horvitz (1999). "Principles of Mixed-Initiative User Interfaces" — CHI '99

Background:

— Allen et al. (1999). "Mixed-Initiative Interaction" — IEEE Intelligent Systems

— Parasuraman et al. (2000). "Types and Levels of Human Interaction with Automation" — IEEE Trans. SMC

Design Frameworks:

— Amershi et al. (2019). "Guidelines for Human-AI Interaction" — CHI '19

— Holmquist (2017). "Intelligence on Tap"

Media Sources

Gmail Smart Compose | Gmail Spam Filter | Google Maps | iOS Focus Modes | Spotify Discover Weekly | Apple Watch Fall Detection | Slack | Autocorrect