Abstract
By embedding information technologies into tangible products a new class of products is created that we call smart products. Smart products use product information in product-centered communication with users. Communication of smart products is handled by dynamic product interfaces. We present a model for QA-based dynamic product interfaces and its implementation DyPI. It is based on a schema-driven question-and-answer approach for Natural Language understanding and generation. Product information is described by web-based semantic representation formats that are stored in distributed repositories. Communication between smart products and users are run on a dedicated middleware (Tip ‘n Tell) that supports user interactions with products by wireless, RFID-based infrastructures and manages requests on product information.

Keywords: ambient intelligence, electronic commerce, natural language systems

1 Introduction
With digital media, product information has exploded. For Internet alone advertising companies spend approximately $16-18 billion in 2006 with about 40% on tangible products, according to Interactive Advertising Bureau (IAB) and PricewaterhouseCoopers (PwC). Product information and tangible products are commonly used in separated situations. For instance, information created for supporting purchase decisions are offered on the Internet and reduce overall search costs (Bakos, 1997) but this information is currently not available in tangible shopping environments. Empirical studies indicate that consumer groups, such as shopping lovers and belonging seekers, would intend to use value-added shopping services in tangible shopping environments that provide, for instance, shopping alerts and product information (Mort & Drennan, 2005).
Within this context, we explore a concept, called smart product, that virtually and physically merges the concepts of tangible products and information products. In our approach this is achieved by a dedicated middleware, called Tip ‘n Tell, which is a technical infrastructure for value-added mobile services that allows embedding of digital product information into tangible products and thus
supporting product mediated communication between products and users (Maass & Filler, 2006). This approach enables new forms of product interfaces by merging information user interface design and industrial product design. In this article we introduce the concept of a question-and-answer-based dynamic product interface for smart products and with DyPI an implementation of this model for shopping domains. We adopt a design science methodology (Hevner, March, Park, & Ram, 2004) and present here our results in the sense of a “proof by construction” (Nunamaker Jr, Chen, & Purdin, 1990). In section 2 we discuss how product information is used in online purchase situations and review basic technical attempts for the use of digital product information in tangible environments. With smart products, the Tip ’n Tell architecture and dynamic product interfaces we introduce basic concepts in section 3. Section 4 reviews approaches of Question-and-Answer-based Natural Language systems, introduces the model of dynamic product interfaces and its implementation DyPI. Finally, we discuss our current results and future work (section 5) followed by a summary (section 6).

2 Product information in purchase decision situations

For both, digital products and tangible products, product interfaces mediate product-centered communication and product experiences between users, producers and vendors. For digital products, product and interface are both implemented by digital representations while product interfaces for tangible products can be implemented in matter or also by information technologies. Product experience and product communication take significant roles in influencing consumer preferences and behaviour (Narayanan, Manchanda, & Chintagunta, 2005). For online shopping situations it is shown that customized online communication can help to attract customers (Ansari & Mela, 2003).

On technology side, some aspects of product interfaces for tangible products are studied under the umbrella of tangible user interfaces (Ishii & Ullmer, 1997). For shopping scenarios, assistant systems have been developed that support dialogs between users and products. The Mobile ShoppingAssistent (MSA) focuses on multimodal communications between a single product and a single user (Wasinger, Stahl, & Krüger, 2003). Earlier systems, such as MyGrocer (Kourouthanassis, Spinellis, Roussos, & Giaglis, 2002) and (Fox, Johanson, Hanrahan, & Winograd, 2000), venture the integration of tangible objects and digital representations. The general technical architecture of these systems is based on wireless communication and object identification technologies that consist of a mobile device, an identification device, a wireless communication infrastructure and
Communication with smart products in shopping situations
appropriate web services that provide relevant information and processing
capabilities (cf. Fig. 1).
In the following section we describe the concept of a smart product and the
supporting Tip ‘n Tell middleware in more detail before we focus on dynamic
product interfaces.

3 Communication with smart products in shopping
situations

3.1 Smart products
The concept of a smart product embeds customized real-time communication into
any kind of tangible product. It extends traditional views on products (Bloch,
1995; Ulrich & Eppinger, 2004) in the sense that they can adapt tangible products
to usage contexts. For smart products we identified three core requirements:

- (R1) adaptation to situational contexts,
- (R2) adaptation to actors that interact with products or product bundles,
  and
- (R3) adaptation to underlying business constraints.

The first requirement places a smart product into a dynamic usage context that is
given by a situation consisting of a set of actors, products, services, entities,
workflows, protocols, and qualities, such as time, space, or emotions, e.g.,
(Schmid, 2000; Schultze & Carte, 2007) and structured by genre systems, e.g.,
(Yates & Orlikowski, 2002).

With the second requirement smart products are either perceived as a tool
(product-as-tool) or as an actor (product-as-actor) itself that provides
communication skills (Schmid, 2000). The ‘product-as-actor’ view assigns
products with anthropomorphic properties (Wasinger & Wahlster, 2006).

Finally, a smart product is required to communicate according to business
constraints, such as business rules, business models, transaction models, and legal
constraints. Examples for business rules are dynamic pricing and bundling
strategies (Robinson & Lakhani, 1975; Truffelli, 2006).

We have refined these requirements by the following operational requirements:

1. Situatedness: recognition of situational and community contexts (R1)
2. Personalisation: tailoring of products according to buyer’s and consumer’s
   needs and affects (R2)
3. Adaptiveness: change product behaviour according to buyer’s and
   consumer’s responses and tasks (R2)
4. Pro-activity: anticipation of user’s plans and intentions (R2)
5. Business-awareness: consideration of business and legal constraints (R3)
6. Network capability: ability to communicate and bundle with other
   products (R3)

3.2 Tip ‘n Tell middleware for smart products
The automatic generation of behaviour of smart products by information
technologies that is compliant with operational requirements depends on dedicated
formal representations of product-related and context-related information. We
have aligned these requirements with a set of six information types, called facets
(cf. tab. 1) that are integrated for each smart product by a *Smart Product Description Object* (SPDO) (Maass & Filler, 2006): (1) product description (PD), (2) presentation description (PRD), (3) community description (CD), (4) business description (BD), (5) trust&security description (TS) and (6) self description (SD).

<table>
<thead>
<tr>
<th>Requirement Facets</th>
<th>Situated-ness</th>
<th>Personalisation</th>
<th>Adaptiveness</th>
<th>Proactivity</th>
<th>Business Awareness</th>
<th>Network Capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td></td>
<td></td>
<td>✳</td>
<td>✳</td>
<td>✳</td>
<td>✳</td>
</tr>
<tr>
<td>PRD</td>
<td></td>
<td></td>
<td>✳</td>
<td>✳</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD</td>
<td>✳</td>
<td></td>
<td></td>
<td></td>
<td>✳</td>
<td></td>
</tr>
<tr>
<td>BD</td>
<td>✳</td>
<td>✳</td>
<td></td>
<td>✳</td>
<td>✳</td>
<td>✳</td>
</tr>
<tr>
<td>TS</td>
<td></td>
<td>✳</td>
<td>✳</td>
<td></td>
<td>✳</td>
<td>✳</td>
</tr>
</tbody>
</table>

*tab. 1: Alignment of requirements and types of information*

On implementation level, each facet is formalised by web-based semantic representations, currently RDF(S) (McBride, 2004), which supports web-based processing of SPDO information (Maass & Filler, 2006).

SPDO information is processed by the Tip ‘n Tell middleware (cf. Fig. 2). If a user requests information about a product in focus this request is sent to the *SPDO broker* via a specialised protocol. In our current implementation we use a SPARQL-based request protocol (Prud’hommeaux & Seaborne, 2007). This broker collects requested product information from web services that are referenced by the product ID in the product description facet of the product’s SPDO. The SPDO broker integrates product information from different sources into SPDO representations (Maass & Filler, 2006) and sends it to the dynamic product interface of the smart product in focus.

In this article, we focus on how adaptiveness can be partially achieved by dialogue-centered communication based on Natural Language technologies. This approach is grounded on the observation that the more natural a product communication the higher is the acceptance of services and information in online scenarios (Bo & Benbasat, 2007; Cho, Kim, & Kim, 2002; Häubl & Trifts, 2000).

![Fig. 2: Tip ‘n Tell architecture](image-url)
3.3 Dynamic product interfaces of smart products

Dynamic product interfaces are a special kind of user interface (Shneiderman, 1997) that exhibits real-time communication at the local point of interaction according to the requirements of smart products. This means that dynamic product interfaces are not only localised in time but also in physical and social spaces. Contemporary dialogue applications provide speech-to-text systems that are successfully used in customer relationship management systems (Aberg & Shahami, 2001). But in shopping situations speech is difficult to use because it interferes with privacy needs of customers. Therefore we propose a schema-directed Natural Language approach that efficiently allows directed composition of questions on products.

In the following section, we describe our schema-approach for dynamic product interfaces. First, we present a review of existing QA systems followed by a categorisation scheme and our derived QA-based approach.

4 A QA-based model for dynamic product interfaces

4.1 Review of Question-and-Answer (QA) systems

According to the discourse theory, discourses of communication is purpose-driven (Grosz & Sidner, 1986), i.e. driven by the intention to achieve a particular communication goal. Communication goals are more efficient to achieve if there exist predefined discourse structures that are mutually agreed upon and stereotyped in social communities, such as shopping situations (Lim, 2002). In those domains, scripts help to efficiently organise understanding and generation of Natural Language communication. Hence, we derive that communication in tangible shopping situations is driven by scripts with pre-defined sets of communication goals (Leigh & McGraw, 1989).

Question-and-Answer (QA) systems are often used in the context of unstructured information sources, such as provided by content management systems or search engines for the WWW (Voorhees, 2003). For instance, McKeown’s TEXT model proposes a schema-driven approach that supports the association of different schemata with different question types (McKeown, 1985). A schema is instantiated by text entities retrieved from a lexicon. QA systems allow requests on information spaces by query languages that are based on subsets of Natural Languages (Li et al., 2002).

We have identified six requirements for dynamic product interfaces of smart products used in shopping environments. First, the Natural Language component has to be executable on mobile devices within web-based infrastructures. Furthermore, contents shall be represented by semantic representation languages and stored in distributed repositories. Next, due to diversity, questions are not preconfigured but dynamically constructed at run-time. Furthermore, response time must be short because of its anticipated strong influence on user acceptance in shopping situations. Finally, the QA system shall support object-centered representations that can be used for product representations. According to these requirements we have reviewed leading QA systems: QASCU (Kosseim, Beaudoin, Keighbadi, & Razmara, 2006), InsunQA06 (Zhao, Xu, Li, & Guan),
QA Chancer (Hickl et al., 2006), Ephyra (Schlaefer, Gieselmann, & Sautter, 2006) and SmartWeb (Reithinger et al., 2005) (cf. tab. 2).

<table>
<thead>
<tr>
<th>Systems/Requirement</th>
<th>QASCU</th>
<th>InsunQA06</th>
<th>Chancer</th>
<th>Ephyra</th>
<th>SmartWeb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution on mobile devices</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Handling of semantic description languages</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>N/A</td>
<td>+</td>
</tr>
<tr>
<td>Joining of information from distributed repositories</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Web-based infrastructure</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Dynamic construction of questions</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Object-centered representations</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**tab. 2: Review of QA systems**

Neither of the reviewed systems meets all requirements. SmartWeb comes closest to our requirements but lacks object-centered representations. Thus, we made it our objective to develop a QA-based dynamic product interface, which satisfies the framework of requirements by consideration of results of the reviewed QA systems.

4.2 **Representation of questions-answer schemata**

In most QA systems, question and answer sets are organised in typologies that define a system’s vocabulary. Some attempts, such as the ISI Question Answer Typology, try to categorise questions for any kind of application by a homogeneous set of questions types (Hovy, Hermjakob, & Ravichandran, 2002). We started with an initial text analysis of a corpus of 80 questions that have been collected from real-life shopping situations which resulted in a concise set of communication goals (based on (Hovy et al., 2002)):

1. **Survey**: a customer requests an overview of products, features, price, guarantee etc.
2. **Factoid**: facts about products, e.g., which MP3 player can change the colour of their display? What is the price of bag?
3. **Quantification**: quantitative information about products, e.g., how many dresses are available in green?
4. **Operations**: information on the usage of a product, e.g., what is the average run-time of MP3 players?
5. **Existence**: information about the existence of a product or product type, e.g., is there a MP3 player with Flash displays?
6. **Definition**: explanation of a particular concept related to a product, e.g., what is the meaning of UMTS?
A QA-based model for dynamic product interfaces

7. **Justification:** argumentation for particular product information, e.g., why is it important to have a Bluetooth interface?
8. **Comparison:** comparison of a product with other products on a particular level of detail, e.g., which other MP3 player is similarly good in their price-performance ratio?

Question types were represented by a semantic web representation format, i.e. RDF(S) (McBride, 2004). The representation by RDF(S) has two advantages: (1) it supports exchanges of semantically annotated data schemas across heterogeneous application infrastructures (Berners-Lee, Hendler, & Lassila, 2001) and (2) offers a basic classification system that can be used for representation of taxonomic relationships. In the following, we describe how questions are incrementally constructed by user-system interactions.

### 4.3 QA-based dynamic product interface

A QA-based dynamic product interface (QA-DPI) assists customers while evaluating a particular product in a shopping situation. It incrementally guides the customer through a schema-based question construction process. Communication goals are derived from question types which are used to generate answers based on retrieved semantically annotated product information.

The architecture of a dynamic product interface in general consists of a mobile device, a dialogue web service and a linguistic knowledge base (cf. Fig. 3). The GUI control maintains the user interface and decodes the user interactions into requests, which are sent by the client system control to the dialogue web service. The dialogue manager processes user input, queries correct knowledge bases and returns results to the mobile device. The modules schema processing, lexicon processing and SPDO Linker are involved in the question formulation. At first, the dialogue manager queries the set of schema with possible QuestionTags. According to the selection of the user one or more schema are activated with pre-configured segments. A scheme is instantiated by incremental processing of each module. The schema processing and the lexicon processing have access to
the linguistic knowledge base, which consists of a schema repository and a lexicon. The SPDO linker interacts with the SPDO broker of the Tip ’n Tell middleware to query the product information in SPDO format. SPDO information is passed by the SPDO linker to the lexicon processing module which uses it for scheme instantiation. Results are returned to the mobile device.

Our model of a QA-DPI is based on a three-parted discourse structure (Grosz & Sidner, 1986), schema-driven communication (McKeown, 1985; Schank & Abelson, 1977) and clue words that support efficient interpretation of sentences (Cohen, 1996). The intentional structure of a shopping discourse is focussed on the intention to receive relevant information about a product or product bundle (Creusen & Schoormans, 2005; Venkatesh, Ramesh, & Massey, 2003; Young Eun & Benbasat, 2004).

4.3.1 Incremental generation of questions

The semantic knowledge base of the QA-DPI consists of (1) SPDO descriptions of the product in focus and associated products and (2) the QA schemas as well as the lexicon, which categorises words and assigns them to meanings (Grishman & Calzolari, 1997). A question contains question objects with communication goals (cf. tab. 3).

Communication goals refer to tree-structured representations whose particular segments are interconnected as a linked list with several types (McKeown, 1985): questionTag, Product, verb, auxiliary, adjective, negation, property, productCategory and particle. For instance, an essay scheme has six possible scheme instantiations (cf. Fig. 4).

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Communication Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supplement/ Operation</td>
<td>Survey</td>
</tr>
<tr>
<td></td>
<td>Quantification</td>
</tr>
<tr>
<td>Essay</td>
<td>Factoid</td>
</tr>
<tr>
<td></td>
<td>Operations</td>
</tr>
<tr>
<td>Decision</td>
<td>Existence</td>
</tr>
<tr>
<td>Definition</td>
<td>Definition</td>
</tr>
<tr>
<td>Justification</td>
<td>Justification</td>
</tr>
<tr>
<td>Comparison</td>
<td>Comparison</td>
</tr>
</tbody>
</table>

*tab. 3: Association of schema and communication goals*

The lexicon taxonomically and ontologically maps the meaning and the content of segments (e.g. property). It enables the QA System to link a segment of a scheme with its meaning and form.

The particular linguistic segments in the lexicon are interconnected by paradigmatic and syntagmatic semantic relations (Mehl, 1993). The assignment of schema and communication goals during the natural language processing is simplified by means of key words (Cohen, 1984), which are organised and linked in the lexicon. A schema instantiation is incrementally constructed by offering next-level segments to the user till a leaf-node is reached.

An essential feature of a QA-based DPI is its mechanism for mapping schema instances on linguistic level with product information given by SPDO descriptions. For this, the lexicon passes segments of the type product,
A QA-based model for dynamic product interfaces

productCategory, person, mediaEvent, medium and shop on to the SPDO broker that matches it with appropriate SPDO entities. Controlled by the dialogue manager of the web service the linguistic knowledge base is temporarily joined with the global SPDO description of the product.

The dialogue manager queries the SPDO description through the component SPDO linker and extracts the particular product information. The SPDO linker returns the results to the dialogue manager, which integrates the specific product information into the instantiated question scheme. Answers are generated accordingly by instantiation of schema.

### 4.3.2 Example

This process becomes apparent in the following example of the extraction of the question “How much costs Green Paradise?” (cf. Fig. 5). At the beginning of the processing all question words of the question schema are listed. The user selects the question word “How much”.

The QA system might retrieve multiple schema. In this case, there is only one applicable scheme – essay -, which is linked to the appropriate communication goal factoid. According to the essay scheme, only a verb can follow.

```
:how_much  a query:factoid;
  query:pattern [ verb:cost [ a <noun:product>;
    spdo:pd-name "?x" ] ] .
```

Verbs are instantiated through lexicon-look-ups. We assume that the user chooses segment “cost”. According to the schema, the next segment must be a product.
The name of the product (“Green Paradise”) is extracted from its SPDO and returned to the QA-DPI, which offers the last segment to the user which might be completed as follows: “How much costs Green Paradise?”.

4.4 Implementation issues on technical level

The QA-DPI model has been implemented as part of the existing Tip ‘n Tell middleware for smart products, see Fig. 2. In our current implementation Tip ‘n Tell shoppers are equipped with a PDA and a RFID-reader pen (Cathexis IDBlue RFID Pen) and smart products are annotated with RFID tags (ISO15693, HF range with 13,56 MHz) which carry URL references to the location where a product’s SPDO is stored.

The whole Tip ‘n Tell architecture is implemented by a web service architecture on the basis of the Jena 2.0 system (jena.sourceforge.net). On mobile client side we use the .Net Compact Framework on PDAs (HP iPAQ Pocket PC). Run-time tests of our Tip ‘n Tell implementation show stable behaviour of all components with exception of the wireless and RFID-based components that are rather slow and error-prone. This problem can be solved by alternative OI technologies such as bar-code scanning (Adelmann, Langheinrich, & Floerkemeier, 2006).

5 Discussion and future work

Our proposed web-based three-tier Tip ‘n Tell architecture consists of (1) a dynamic product interface (2) a SPDO broker and (3) product information repositories. This allows coping with dynamically changing product information sources from different providers. The SPDO broker integrates distributed product information and translates heterogeneous data into semantically annotated SPDO formats (Maass & Filler, 2006). This works fine for managed product information sources but provides typical ontology matching problems for product information based on unknown data schemas (Doan, Madhavan, Dhamankar, Domingos, & Halevy, 2003).

For QA-based DPI, we have focused on a proper design of QA schemas, appropriate selection and information matching mechanism. Even though that the surface generation mechanism is simple, first evaluations with test persons indicate that probands perceive it as very helpful. In informal pre-tests probands effortlessly constructed questions on the mobile device in shopping situations, navigated the product information space and rated this system as beneficial. Next we will conduct formal behavioural studies on the role of smart products in purchase decisions similar to (Häubl & Murray, 2003).

QA-DyPI support users in their navigation in dynamically constructed product information spaces. Currently we work on regeneration mechanisms that prevent separation of product information spaces from tangible products that are present in a particular context. This will allow restricting information to products, for instance, only those that are available in a particular shop or mall.

In respect to operational requirements for smart products, QA-based DPI contribute to the adaptiveness requirement by offering a question and answer dialogue service to a user. In more complex situations it will be required to track and aggregate multiple goals into plans that represent more complex user intentions, e.g., (Reithinger et al., 2003). Obviously dynamic product interfaces in general are natural mediating services for mechanisms that implement the
Summary

personalization and pro-activity requirements by mechanisms for maintenance of user models, analysis of user activities and derivation of user intentions and plans.

6 Summary

Mobile commerce provides opportunities to sell products and services at any time and any location. This will create more contact points with customers and improve customer services by bringing products and services directly to customers (Frolick & Lei-da-Chen, 2004). Hence, this requires product interfaces that can cope with these business requirements, i.e., being situated, personalised, adaptive, proactive, business-aware and network capable. With DyPI we have implemented the QA-DPI model into the Tip 'n Tell system. Thus, a first step towards a full implementation of the smart product concept has been achieved.

References


Summary


Zhao, Y., Xu, Z., Li, P., & Guan, Y. InsunQA06 on QA track of TREC2006, Fifteenth Text Retrieval Conference (TREC 2006).