Can Context Tame Web TV?

Reconciling infinite choice and zero effort

Venu Vasudevan, Jehan Wickramasuriya, Silviu Chiricescu
Betaworks, Applied Research Center
Motorola Mobility, Libertyville, IL

Gilles Drieu
Advanced Software & Services
Motorola Mobility
Sunnyvale, CA

***Abstract*— Web TV is making a comeback, as users increasingly *co-consume* TV related web content on their companion devices (tablets and smartphones) in sync with TV watching. In this 'dual screen' experience, users download and invoke TV apps on their companion devices to personalize their Superbowl or American Idol TV experience with a web counterpart. Unfortunately, the added effort in continually finding and invoking one of many TV Apps threatens to undermine the lightweightness of cognitive experience that makes TV addictive. In this paper, we propose a 2nd screen *contextual UI* that balances the richness of an App driven 2nd screen experience with the simplicity of a desirable TV experience. We further share user feedback that is both reinforcing and surprising in relation to the appropriate and inappropriate uses of context in a living room setting.**

Keywords—Search & discovery, multi-app interactivity, media multi-tasking, context-aware, development tools

I. INTRODUCTION

There is a resurgence in Interactive TV, thanks to an increasing tendency amongst TV viewers to watch TV with a 2nd screen device (e.g. tablet, Smartphone, laptop) in hand. Some studies [1,3] estimate that over 80% of TV viewers have a 2nd screen on hand while watching TV. These and other studies [2] point out that not only is the 2nd screen used for TV-related activities, but that we are beginning to understand the specific transition points [2] at which a user switches attention from the TV to the 2nd screen (and back). We view the 2nd screen as a first step in the ***dispersion*** of a TV experience across living room devices, with accessories as a potential 3rd screen in the future.

The dispersion of an interactive TV experience is accompanied by the ***atomization*** of TV viewer interactivity into applications (*Apps)*. In contrast to the all-in-one integrated TV platforms of the past, the new 'Web TV' is emerging as a collection of third-party *TV Apps*, mimicking the mobile ecosystem. These Apps are developed and deployed into the marketplaces (iOS and Android) by both existing content players and new social ones. A notable proportion of these TV apps are *native wrappers* around TV related web content – thus componentizing the web for a TV experience.

TV apps come in two flavors depending on whether the app creators are incumbents or new market entrants. Incumbent celebrities (e.g. Oprah) and content studios (e.g. HBO) use apps to convert an indirect connection with their viewers into a direct relationship, thus creating community around their content. New social players such as GoMiso and other 'TV Check-in' players use the opposite path – leveraging their user base to create proprietary user-generated content (UGC) around their community [1]. The Transmedia Web associated with TV is thus fragmented into a plethora of branded apps that have to be searched, discovered, downloaded, provisioned, invoked and managed, concurrently with TV watching. Industry pundit GigaOM predicts that the population of such TV Apps to be in the tens of thousands by 2015 [2]. Our own informal search of the Android marketplace indicates that search terms relating to TV returned over 1500 results in early 2012 (up from about 250 in early 2011).

While the emerging *TV App* ecosystem adds an exciting richness to the TV experience, it also threatens the very attributes that make TV attractive as a medium. As pointed out in a TV-related user study [7], the key desirable attributes of TV that users value is that it 'just works', is instant ON and isn't cognitively demanding. Even the current population of applications can cumulatively weigh down the TV experience if all the above-mentioned app operations have to be manually invoked by the user. The key challenge being addressed in this paper is: *how can we marry the rich web of TV Apps with the simplicity of a TV experience, into a composite that is both rich and effortless*.

The problem of a cognitively overloaded 2nd screen TV interface has two components. First is the increased user effort in *searching*, provisioning and contextually selecting the right TV app(s) for a particular live TV program that the user is watching. Secondly and given that you have the right set of Apps - TV viewers face the problem of *interacting* with a large set of apps on their tablets while simultaneously watching TV programming on the big screen. We believe

978-1-4673-5077-8/13/$31.00 ©2013 IEEE

that a focused and disciplined use of context-based adaptation could be key to maintaining effortless interfaces in the face of an onslaught of TV Apps. ***Morphix***, our context-driven TV UI, attempts to solve the search & discovery problem by leveraging ***media context***, and the interactivity problem by adapting to ***device context***.

Morphix addresses the search & discovery problem by auto-discovering and auto-installing the right set of apps for a program that the user is watching. The key challenges in this aspect of the interface are *multi-tenancy* and *re-entrancy*. *Multi-tenancy* means that the 2nd screen experience is a curated experience that is jointly owned by content studios, cable operators, broadcasters, end users and other third parties (e.g. sports leagues such as the NFL or UEFA). *Re-entrancy* refers to the fact that a 2nd screen interface needs to save and restore channel state, as a user flips away from (and back to) a particular program. Our solution to multi-app interactivity shares much with other work on contextual interfaces, except that we aim to solve the problem more narrowly and deeply in the realm of context conditions that are relevant to the living room couch (e.g. position of your 2nd screen device, the ambient light level in the room).

The next two sections describe the architecture of the *Morphix* App container. Subsequent sections describe the actual prototype, preliminary results from a user study on the efficacy of such a 2nd screen interface, and conclude with ongoing and future work.

II. MORPHIX : A CONTAINER FOR APPIFIED TV

Core to our view of how Appified TV [9] will evolve is the *separation of apps and containers*. Apps will continue to be developed by app developers to support both the ability for dual-screen TV watching as well as a user interface that works as a standalone entity. A smaller number of *container developers* will develop app containers that transparently pass useful pieces of context to contained apps (e.g. TV *metadata* such as channel info, and potentially TV *data* such as a clip of the current TV scene). Containers will thus enhance the experience provided by the app that is included and invoked within the context of the container. From a business perspective, app developers will be part of a number of small innovative companies, while container development will be the purview of large companies with marketing muscle (e.g. major content studios and TV distributors) and access to large numbers of subscriber eyeballs. When multiple TV apps are invoked on a tablet in the context of a TV program, the app container runtime will have to provide support to deal with certain contextual app management and save/restore style lifecycle operations.

To address these concerns, we built a prototype container for mobile 2nd screen devices called Morphix, to understand the issues in Appified TV ahead of the availability of such a capability in the industry. The purpose of this container was to expose contextual capabilities to TV applications in a consistent and easy-to-use way, as well as provide a level of control for stakeholders (e.g., content studios, distributors etc.) to customize it as needed. Figure 1 is a wireframe that outlines the various UI elements present in Morphix. While this resembles a traditional window manager (or browser) in many ways, the key differences are that the app selection is contextual to the TV content, and that runtime app management is context-sensitive to the TV companion (device) modality. These two points are described in more detail in the next two sections.

III. APPLICATION SEARCH & DISCOVERY

With both the Apple and Android marketplaces surpassing 750K apps each, search and discovery has become a bottleneck. This has led to notable startup activity in the app finder space as it pertains to standalone apps. In the standalone app space, it is assumed that once a user has discovered an app (i.e. is aware of it and makes the effort of downloading it to his device), the task of discovering that app is complete. TV app discovery shares that part with standalone apps, but has additional aspects that are a function of the fact that a TV app is supplementary and sometimes synced to TV content.

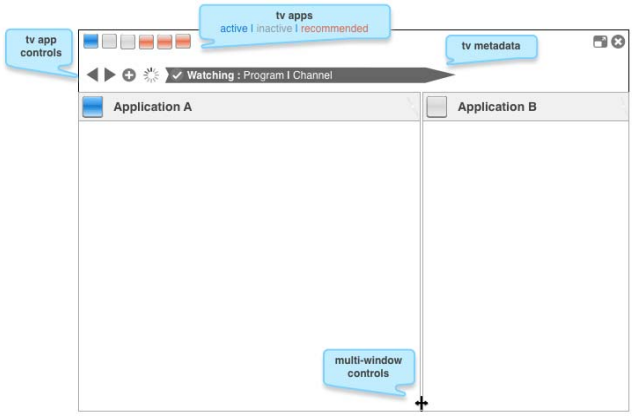

Figure 1: Morphix prototype wireframe

If one considers "app discovery" from a user-centric perspective (just in time recommendation of the right apps to a user in the right TV context), then TV app discovery has the following special requirements.

• *Time-bound discovery*. TV apps have to be exposed to the user in a way that is appropriately bound to a program (or part thereof), episode or cast without overt effort by the user to find the apps. A single TV program (e.g. ESPN SportsCenter) might have multiple associated apps (e.g.

Twitter and ESPN Fantasy Football) and conversely a single app (e.g. Twitter) might map to multiple programs (e.g. ESPN & American Idol). Certain apps (e.g. ESPN March Madness Fantasy Basketball) might be ephemeral and event-specific, or regional and personal (e.g. a Chicago Bears app), adding contextual elements to the program-app mapping.

• *Parameterized reentrancy*. Given that TV Apps are supplementary and ideally synced to TV content, we expect a number of them to function with real-time TV content. The simplest current example of such parameterization is a number of Twitter-based apps that work off program specific hashtags and keywords to filter content that is specific to a TV program. We anticipate more sophisticated apps such as personal telestrators (i.e. TV image annotators) or "advergames" that will leverage images, word sequences or utterances. Such content-aware TV apps require a means to request and receive the requisite content (possibly via a licensed API), since all TV content is subject to digital rights management (DRM). Additionally (to the point of reentrancy), TV apps have to save and restore their dialog state as users flip out of a TV program and back in, so that the app experience has continuity much as the TV content experience does.

• *Divided ownership*. "First screen" content is generally jointly owned by the content studio, a distributor (e.g. a cable/satellite operator), (potentially) another content creation entity (e.g. sports league) and sometimes other entities (e.g. major media personalities such as Oprah Winfrey). These business entities view 2nd screen content as an extension of the TV experience. There is some validity to this in that a) the core applications for a show are often created by one of these entities and b) a compelling TV App ecosystem relies on access to 1st screen content owned by one or more of these businesses. Thus the TV App suggestion interface has an organic element that might rely on app download trends on social (e.g. buzz) indicators, but also has a significant sponsored component that is driven by 2nd screen apps that 1st screen content owners feel to be additive to the TV experience.

To accommodate these three requirements, we developed augmented the Morphix container with a prototype TV app search and discovery interface. The interface is modeled as a contextual App dashboard with a finite set of "app slots" for any program the user tunes into. The fixed set of app slots bounds the cognitive load on 2nd screen interactivity, and the fact that these are auto-suggested (based on either sponsored or organic cues) means that no effort is required on the user's side to locate program-related apps. The app slots' ownership is shared by the business entities associated with the show (one can consider the user as one such entity). The mapping of apps to a program is specified via an App Campaign Manager (described in [9]), within which the owning entity for an app slot can author campaign rules (similar to how advertising campaign mangers operate). Campaign rules in this case enable location, device and user specific targeting of apps within an app slot owned by a content owner. The source of TV (meta)data described in the parameterized reentrancy bullet point varies with TV technology (broadcast vs. over-the-top TV vs. IPTV vs. cable), ranging from automated content recognition to TV backoffice integration. Our companion app container provides an interface to deliver this data to content-aware apps in a standardized manner, and we augment this premise in the following sections.

IV. MULTI-APP INTERACTIVITY

The previous section dealt with appropriate *just-in-time* exposure of content-relevant apps to users. In contrast, this section addresses the challenges in managing multiple "live" app sessions relevant to a TV program in a living room setting. This requires the capability to create a tablet interface that supports a desktop-like capability of hosting multiple applications in multiple separate windows. Additionally, the runtime system needs to manage each TV App's run-time state and presentation style to both channel changes and changes in device affordance.

On the first point - while the multi-window capability on a TV-centric UI resembles a PC desktop in some ways, the *lean-back* nature of the UI motivates a higher degree of automation in window management vs. a lean forward desktop UI. To this end, our container, *Morphix* makes a number of (customizable) UI decisions for the user – the location and UI real estate for invoked apps as a function of the number of open apps, the location in which a new app will open, and the finite set of choices for App window sizes. Additionally, *multi-tenancy* suggests that app developers be given control of UI configurations in which their app simply won't function well, and therefore shouldn't be auto-invoked. To this end, we enable app developers to define minimum constraints for their apps to function properly, and include these in UI automation decisions.

On the second topic of contextually adaptive UI's, the needs of a TV UI again are a hybrid between contextually aware desktop UIs (such as Apple's [12] and contextually aware mobile UIs [10]. TV UI's resemble desktop UI's in that much of the needed adaptation is to *indoor* context changes (e.g. noise and lighting) as well as changes in the number of active applications. Unlike desktop UI's, their TV counterparts also need to adjust to changes in *user affordances* (e.g. sitting up vs. recumbent) and changes in the level of their attention. TV UI's resemble their mobile counterparts in that interaction on both is an 'expensive' user operation, but TV UI's differ in needing to deal

primarily with a bounded set of living room context changes, as opposed to outdoor contexts associated with pervasive mobility.

With these points in mind, our architecture for an adaptive TV UI had the following elements:

- Integration with a (device) context sensing layer (in our case, Motorola Smart Actions [14]) to support a bounded living room context information model
- Providing developers with a simple rule based language to create Morphix *adaptation wrappers* for their TV apps
- The ability for container level overrides of some or all of these adaptation wrapper specifications to create coherent adaptations across multiple Apps on a single rendering
- Emulation capabilities to test out the behavior of adaptations in real and synthetic context conditions (e.g. the ability to record and replay a user's living room on Superbowl Sunday)

We deliberately chose simplicity and "good enough" over expressive power in many of the above design choices. For instance, we chose rule-based specifications over full function constraint solvers [4,5,6] because we felt that the non-determinism introduced by the latter might not work well for a consumer electronics interface. We tried to conform in our language constructs to CSS3 analogs (and something with the flavor of a substantially enhanced CSS3 Media Queries form), as both familiar notation to web developers and an adequate substrate to build TV UI enhancements upon. We took a "tools over edicts" view in relation to the debate on the pros and cons of auto-adaptive vs. user customizable vs. hardwired UIs [6,8]. By providing context emulation tools, we hope to enable people to create effective contextual interfaces, and leave the choice of whether to auto-adapt to the best judgment of user experience designers.

V. PROTOTYPE

The prototype we built supports inter-program navigation component and app experiences. The former is an enhancement to the concept of an EPG (Electronic Program Guide) that is familiar to TV users, but with numerous social features (depicted in Figure 2). The latter is the 2nd screen experience that a user sees when (s)he is *within* a program, as described earlier in Figure 1. Once a user tunes into a channel via the enhanced EPG, *Morphix* provides for three main capabilities for the 2nd screen app experience. These include (a) app discovery & search capabilities for TV apps, (b) layout management for hosting multiple concurrent applications and (c) contextual adaptation for apps that utilize on-device sensing and other contextual cues. App discovery is supported by a campaign manager interface (as previously described in Section 2), where both content owners and users can suggest appropriate candidate apps that are matched to the content and user.

While our long-term goal is to support a mix of Android and Web Apps, the current prototype is limited to Web Apps so as to get a preliminary sense of the efficacy of the framework. Although we have looked into solutions to support native Android applications for the purposes of layout management capabilities [15], we made a conscious decision to stay away from framework modifications to the underlying operating system (in this case Android). It should be noted that the operating system can inherently provide (via supported APIs) many attractive features that facilitate much of the functionality we built into *Morphix* – namely support for application search and discovery, as well as the contextual adaptation hooks into lower level device capabilities. Support for layout management and contextual adaptation (capabilities (b) and (c) above) involve integration with and layering on context middleware which is described below.

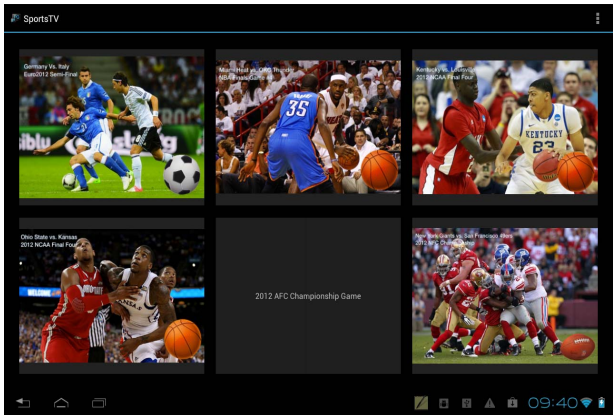

Figure 2: EPG concept application prototype

Web Apps are wrapped within an Android construct called a *Webview*. This Android View around Web Apps enables them to interface with the context support layer [14] on an Android Smartphone. A *Webview* looks exactly like a browser to the hosted Web App, and exactly like an Android component to the Android framework. By wrapping Web Apps in an Android Webview, we can leverage Motorola's context sensing and aggregation framework for Android, which drives the Smart Actions application on mobile devices for intelligent management of sensor hardware.

Atop the device's context middleware, *Morphix* provides a two-way interface between (wrapped) Web Apps and the underlying device context middleware. It translates the App

adaptation wrappers into lower-level context subscription requests supported by the middleware, and conversely low-level context changes into quantized values as required by adaptation wrappers. *Morphix* then executes the actions suggested by the adaptation wrapper, potentially changing the font or pane sizes of the Web App based on the wrapper specification. At a macro level, Morphix performs layout management for hosting multiple concurrent applications, and reconciles conflicting actions of adaptation wrappers based on container level policies.

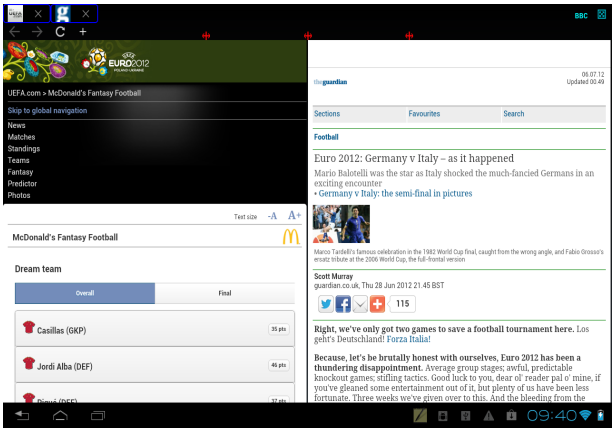

Figure 3: Morphix container displaying two sports applications in concert with a soccer game

Figures 2 and 3 show the prototype customized to Sports TV. Figure 2 shows the enhanced program guide offering viewers the choice of dialing in to one of many concurrently running TV programs. Figure 3 shows the sample App interface for a user who navigates to a particular soccer game (being broadcast on TV). *Morphix* displays 2 panes, each running a Web App (suitably matched to the current content), and each of which support independent adaptation wrappers that respond to changes in both layout and environmental context. As mentioned earlier, both the number of concurrent and active Apps, as well as the forms of adaptation are customized at a per-App level and harmonized at the container level. This prototype configuration was used in a study on the value of this 2nd screen interface to Sports TV fans. This study is covered in more detail in the next section.

VI. Does Context Help or Hurt TV? – Qualitative User Feedback

We solicited feedback from twelve users on various aspects of ***Morphix*** as it pertained to enhancing a TV experience. The user group consisted of a mix of TV viewers with a passion for Sports, and App developers with an interest in the mobile media space. We chose Sports as the domain because it is both naturally interactive as a TV genre, and extremely demanding on the quality and performance of the interactive media experience. The screenshots in Figures 2 and 3 outline the user experience – comprising of a visual sports program guide (Figure 2) that offers a selection of programs, and leads to a customized multi-app UI that is program-specific (Figure 3).

We wanted get user perspectives on three dimensions of a 2nd screen TV experience – *utility*, *design* and *dynamics*. By *utility*, we were looking for specific patterns of use in terms of the coupling of content and cadence across TV program and 2nd screen information feed. In terms of *design*, we were looking for "do's and don'ts" when it came to a rich, multi-paned tablet UI that was also meant to be effortless. And finally, while there are numerous studies about the appropriate use of context in App design, we *specifically* wanted to understand the *dynamics* of a time-variant UI from the perspective of a sedentary user on a couch. *Personalization* is an optimization that we recognize as being integral to the user experience and provides the glue that make the context-triggered adaptations of applications resonate with users. However, our goal with this initial user study was to understand the efficacy of our core system, so we concentrated on the other dimensions.

Utility: what do people do on the second screen?

The intuition going into the study was that 2nd screen was an *engagement amplifier* for the TV program playing on the first screen. So a TV viewer watching a sitcom, reality show or sports program would have an associated interactive experience on the 2nd screen dealing information on actors, player stats, 'directors cut' snippets and so on that would deepen the engagement with the program on the 1st screen.

The user feedback also brought in the notion of *contra-engagement* – that is to say the 2nd screen as a means to stay in touch with programs you are **not** currently watching. In hindsight this makes sense as a particularly useful metaphor in Sports, where a user is trying to juggle multiple games. The fact that the balance between engagement and contra-engagement is a very personal (and genre specific) choice, is pertinent to the discussion about design that follows.

Design: Where is the balance of cognitive simplicity and expressive power

From a design perspective, we wanted to understand the balance of simplicity versus expressivity in terms of both *information* and *interaction*. On information, the question boiled down to how complex is too complex for a supplementary TV user interface? And on interaction, how many customization knobs are too many for the user to bother with?

On the first question, users were almost evenly split on needing somewhere between 2 to 4 apps for a satisfactory 2nd

screen Sports experience (and therefore 2 to 4 panes). Because a user's proportion of contra-engagement varied, the choice of constituent TV apps was very personal. Users were therefore emphatic on *App dashboard customization* – i.e. that they be allowed to pick their companion apps for any TV program. From an *interaction* perspective, users wanted the system to make certain choices for them (e.g. fixed pane size choices, automatic placement of newly invoked app), but have the ability to permanently override these choices (e.g. move a newly invoked App to a more convenient location)

Adaptation: Friend or Foe?

We wanted to get a sense for the relative value of two forms of contextual adaptation provided by *Morphix – media adaptation* and *device adaptation*. The former related to the fact that the *Morphix* "dashboard" changed with the media (TV program) that was playing on the 1st screen, and was re-entrant as users flipped to (and back to) programs. Device adaptation related to automated changes to the *Morphix* user interface in response to device affordances. User feedback on media adaptation was surprisingly positive, with some users calling it "magical". While users strongly emphasized the need for viewer control of companion Apps, they appreciated that fact that the apps were auto-invoked, and maintained their state even if the user left a program and came back to it.

Equally surprising was the *strong negative* reaction to device adaptation. Users felt that the system was highly likely to misjudge (and adapt to) sudden and temporary movements on the couch such as dropping the device, leaning towards a friend, or just getting more comfortable on the couch. It is possible that prolonged use of the interface, along with explanatory interfaces [11] might engender both familiarity and trust in contextual device adaptation. However, the one insight we took away even from this preliminary study was that a couch was a compressed physical space with sudden and exaggerated physical transitions (e.g. people movement, lighting changes, changes in ambient noise). This might imply that device adaptation in living room (and more generally digital home) situations may have some unique needs that bear careful consideration.

VII. CONCLUSIONS

In this paper, we outlined an approach to enriching a TV experience through an app marketplace while retaining the lightweight nature of TV as a medium. To this end, we outlined an architecture that caters to *dual-screen interactive TV* in a manner that balances both user benefit and business architecture. On the former, we investigated how contextual App management might provide TV viewers a highly customized 2nd screen interface that was nevertheless low-effort in both cognition and manipulation. On the latter, we built an interface model that managed multi-tenancy, and took the views of content investors (studios and operators) as wells as App developers into account.

On the usability side - we conducted some preliminary studies on the efficacy of this interface from both a utility and design perspective. The results of the study quantified intuition in terms of the media experience, but uncovered some notable and counter-intuitive results in terms of modality of usage. Future work involves refining both technology and user insights. On the former, we anticipate further user studies that expose nuances of the 2nd screen experience. On the technology front, we're looking to refine platform technology to support faster exploration for app and experience developers, and a higher performance of TV viewers.

REFERENCES

[1] TV Apps: Evolution from Novelty to Mainstream. In GigaOm Research Report, March 2011

[2] "The New Multi-screen World: Understanding Cross-Platform Consumer Behavior," Google - http://goo.gl/XDCbP

[3] Yahoo | Razorfish study on Social TV. http://goo.gl/skfro

[4] G. Badros, A. Borning, and P. Stuckey. "The Cassowary Linear Arithmetic Constraint Solving Algorithm," In ACM Transactions on Computer Human Interaction, December 2001.

[5] S. Feuerstack, M. Bluemendorf, V. Schwartze, and S. Albayrak. "Model-based Layout Generation," AVI, CHI 2008.

[6] K. Gajos, J. Wobbrock, and D. Weld. "Improving the performance of motor-impaired users with automatically generated, ability-based interfaces," CHI 2008

[7] H. Holtzman. "Future of TV," In MIT Media Lab Workshop, 2011.

[8] J. Mitchell and B. Shneiderman. "Dynamic versus static menus: an exploratory comparison," ACM SIGCHI Bulletin, 1989.

[9] V. Vasudevan and J. Wickramasuriya. "Social & Interactive TV: An outside-in approach," Future TV, EUROITV, 2011.

[10] S.K. Kane, J.O. Wobbrock, and I.E. Smith, "Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces," MobileHCI '08 ACM, New York, NY, USA, 109-118.

[11] Lim, B.Y., Dey, A.K., and Avrahami, D. "Why and why not explanations improve the intelligibility of context-aware intelligent systems," CHI '09

[12] Zeidler, C., Lutteroth, C., and Weber, G. "Constraint solving for beautiful user interfaces: how solving strategies support layout aesthetics." CHINZ '12

[13] Mitchell, J., and Shneiderman, B. "Dynamic versus static menu: an exploratory comparison," ACM SIGCHI Bulletin, 1989.

[14] Motorola Mobility SmartActions. http://goo.gl/Fkzt7

[15] Cornerstone, OnSkreen. http://goo.gl/FqLzM