7. Guilford, J. P. (1950). Creativity. American Psychologist, 5(9), 444–454. https://doi.org/10.1037/h0063487
8. Харарі, Ю. Н. (2023). 21 урок для 21 століття. Перекл. Демянчук О. Букшеф, 416 с.
9. Шандрук, С.К. (2015). Психологія професійних творчих здібностей: [монографія]. Тернопіль: Економічна думка, 357 с.
10. Brand, P. A. (2020) Cognitive Semiotics. Signs, Mind, Meaning. London, New York, Oxford, New Delhi, Sydney: Bloomsbury Academic, 288 p.
11. McLuhan, M. (1968). War and peace in the global village. New York: Bantam Books, 190 p.
12. Peirce, Ch. S. (1908). A Neglected Argument for the Reality of God. The Essential Peirce: Selected Philosophical Writings. Volume 2. (pp. 434–450). Ed. by the Peirce Edition Project. N. Houser et al. Bloomington, IN: Indiana University Press, 1998. 650 p.
13. The Psychology Podcast: Right Brain, Creativity, and Meaning in Life w/ Iain McGilchrist. https://open.spotify.com/episode/0FXvc4IQI1wOd2mlRqyovQ
14. Speaking of Psychology: Creativity, insight, and "eureka moments", with John Kounios, PhD, professor of psychology. https://open.spotify.com/episode/0qfXuGpztYVmtXpNGWlxnz
15. Speaking of Psychology: The neuroscience of creativity, with Rex E. Jung, PhD.https://open.spotify.com/episode/4ArMowEfV2d1qXLz4CQE7N
16. Jungian Life Podcast: "Lucid Dreaming: How to Make the Most of a Magical Opportunity". https://open.spotify.com/episode/0ekn8jqdxTO6kuoVbG0fBq
17. The Thinking Abyss: The Neuroscience of Creativity: Where Original Ideas Actually Come.https://open.spotify.com/episode/0KirxybOxh7jn8wtQGBcn

Vladyslav Neduzhko
Kyiv National University of Technologies and Design
Kyiv, Ukraine
Liudmiya Roienko
Kyiv National University of Technologies and Design
Kyiv, Ukraine

ENGINEERING FOUNDATIONS OF SOFTWARE INTERNATIONALIZATION AND LOCALIZATION

Today, software is built for a global audience. This means applications must be adapted for users from different countries and cultures. This process is divided into two main parts: internationalization (i18n) and localization (l10n). Internationalization is the engineering work that prepares the software code to support multiple languages and data formats. Localization is the actual process of translating text and adapting cultural elements, such as dates, times, and currencies. In modern software engineering,

internationalization is not a feature you add at the end of a project. It must be a core part of the system's architecture from the very beginning. This paper explores the fundamental engineering practices needed to build scalable, localized software, focusing specifically on user interface (UI) architecture and search engine optimization (SEO) for web platforms.

Creating software for different regions is an economic necessity. Companies want to reach users everywhere, which requires supporting many languages simultaneously. However, this creates a major technical challenge. Developers must build systems that can change completely depending on the user's location, without breaking the application's layout or slowing down its performance. This fits perfectly into the modern challenge of digitalization: building flexible technology that connects a global society while maintaining a high-quality user experience for everyone.

One of the hardest parts of localization is making sure the User Interface looks correct in every language. A common problem is "text expansion." For example, a short English word like "Save" might translate into a much longer word in German or Ukrainian. If the UI is built strictly for English string lengths, the longer translated words will break the layout, overlap with other elements, or get cut off entirely.

Modern software engineering uses a component-based approach to solve this. Whether developers are using desktop patterns like MVVM or web libraries like React, the best practice is strict decoupling. This means separating the visual UI design from the actual text data. Instead of hardcoding the word "Save" into the program, the developer uses a reference key, such as button.save. When the application runs, it looks at the user's language setting and loads the correct translation file to fill in that key. Because components are reusable, fixing a localization bug in one button automatically fixes it everywhere that button is used.

Despite the benefits of decoupling, traditional localization workflows have a major flaw. Developers and UI designers cannot see how the translated text will look until the

application is fully compiled and running. This creates a slow process of guessing, compiling, and fixing layout errors.

To fix this, engineering systems need "design-time support." Research on dynamically extending UI components shows that developers can add localization logic directly into their visual design tools (Piippo, 2010). This means that while a developer is building a UI component, they can switch between languages right in their code editor or visual constructor. The UI components dynamically change their size and layout to fit the translated text on the screen immediately. This dynamic extension allows developers to catch layout errors early. It saves a significant amount of time because developers do not have to guess if a translated word will fit, creating a much smoother and faster workflow.

When building web applications, localization involves more than just changing the text on the screen. It also involves making sure that search engines, like Google, can find and understand the translated pages. If a localized web application is not engineered correctly, search engines will only index the default language, meaning users in other countries will not find the application when searching in their native language.

This requires a technical strategy called Programmatic Search Engine Optimization (pSEO). According to research on localized web applications, such as online marketplaces, pSEO involves building an architecture that automatically generates highly optimized web pages for different search terms and regions (Hakala, n.d.).

To achieve this, the web architecture must support dynamic, localized routing. The application needs to generate unique web addresses (URLs) for different languages, such as /en/market for English users and /uk/market for Ukrainian users. It is also critical to use server-side rendering (SSR) or static site generation (SSG). If translations are only loaded on the user's device using JavaScript (client-side rendering), search engine bots might read an empty page. By rendering the localized HTML on the server before sending it to the user, the system guarantees that search engines can read the translated text. Furthermore, the application must automatically generate hidden code, like

hreflang meta tags, to tell search engines exactly which language and region each page is designed for.

As software projects grow, managing thousands of translated words across multiple platforms becomes very difficult. A strong engineering foundation requires automated systems. Modern developers use Continuous Integration and Continuous Deployment (CI/CD) pipelines to connect their code directly to Translation Management Systems.

When a developer creates a new UI element and adds a new text key—often using strongly-typed code like TypeScript interfaces to prevent spelling mistakes—the system automatically sends this new key to the translators. Once the translators finish their work, the new text is automatically downloaded back into the application's code base. This process is called "continuous localization." It ensures that the software is always fully translated and ready to be released to international markets quickly and safely, without waiting for manual file updates.

Building software for a global audience is a complex engineering task that requires careful planning. It requires much more than just swapping out a dictionary of translated words. By engineering UI components that can adapt to different languages during the design phase, developers can prevent visual errors before they happen. By building web architectures that support programmatic SEO and localized routing, engineers ensure that their applications are visible worldwide. Using automated tools and clear architectural patterns ensures that modern software is easy to maintain, looks professional in any language, and can successfully meet the demands of a globalized digital economy.

REFERENCES

1. Marco Grönberg User Interface Localization with Design-Time Support by Dynamically Extending User Inter- face Components. Oppimiskeskus | Aalto-yliopisto. URL: (http://lib.tkk.fi/Dipl/2010/urn100217.pdf) (date of access: 29.04.2026).
2. Teemu Kerkola Programmatic Search Engine Optimization in Localized Web Applications: A Case Study on a Firewood Marketplace App. UTUPub - Turun yliopiston julkaisuarkisto |

Etusivu. URL: (https://www.utupub.fi/server/api/core/bitstreams/182b6328-e190-4234-8cea-fc538824ce8b/content) (date of access: 29.04.2026).

Anastasia Korchinska
Kyiv National University of Technologies and Design
(Kyiv, Ukraine)
Nataliia Liubymova
Kyiv National University of Technologies and Design
(Kyiv, Ukraine)

MIN POP CULTURAL PHENOMENA IN THE EUROPEAN SPACE: ADAPTATION STRATEGIES AND CULTURAL DIFFUSION

Summary : This study analyzes the mechanisms of expansion of East Asian pop culture products (primarily Japanese and South Korean) into the European socio-cultural space. The role of targeted public policy ("Cool Japan", "Hallyu Act") and global digital platforms in overcoming geographical and language barriers is highlighted. The influence of these phenomena on the transformation of consumer habits in Europe, particularly in the fields of high fashion, gastronomy and higher education, as well as the role of decentralized fan communities as new subjects of social influence and digital activism, is analyzed.

The modern globalization process is characterized by a departure from the dominance of the Western-centric model and the formation of a polycentric "global mélange", where East Asian pop-cultural phenomena (Japanese anime, South Korean Hallyu "wave", Chinese video games) have become the leading determinants of cultural diffusion in the European space. The theoretical basis of this expansion is the concept of cultural hybridization, according to which Asian media content does not simply copy Western samples, but synthesizes them with local values, creating a "third space" of meanings available for perception by the European consumer.

Cultural diffusion is realized through the combination of media channels and the activity of diasporic communities, which contributes to transculturalism and the formation of new identities among European youth. The success of East Asia in the European market was the result of a purposeful state strategy of "soft power" of Japan and South Korea, which integrated cultural exports (Cool Japan, Hallyu Act) into the system of national branding.

The key adaptation strategies are deep localization and transcreation, which include not only linguistic translation, but also the adaptation of visual symbols and marketing narratives under the